773

PATENTS

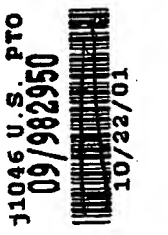
IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of

Junji TAKAGI

Serial No. ~~unknown~~ 09/982,950

Filed herewith



STUB SEARCH LOADING SYSTEM AND METHOD, SERVER APPARATUS, CLIENT APPARATUS, AND COMPUTER-READABLE RECORDING MEDIUM

CLAIM FOR FOREIGN PRIORITY UNDER 35 U.S.C. 119 AND SUBMISSION OF PRIORITY DOCUMENT

Assistant Commissioner for Patents

Washington, D.C. 20231

Sir:

Attached hereto is a certified copy of applicant's corresponding patent application filed in Japan, on 23 October 2000, under No. 322269/2000.

Applicant herewith claims the benefit of the priority filing date of the above-identified application for the above-entitled U.S. application under the provisions of 35 U.S.C. 119.

Respectfully submitted,

YOUNG & THOMPSON

By Benoît Castel

Benoît Castel
Attorney for Applicant
Registration No. 35,041
Customer No. 00466
745 South 23rd Street
Arlington, VA 22202
Telephone: 703/521-2297

October 22, 2001

P14264-A

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日
Date of Application: ２０００年１０月２３日

出願番号
Application Number: 特願２０００－３２２６９

出願人
Applicant(s): 日本電気株式会社

CERTIFIED COPY OF
PRIORITY DOCUMENT

２００１年　８月３１日

特許庁長官
Commissioner,
Japan Patent Office 及川耕造

出証番号　出証特２００１－３０７７５４８

【書類名】　特許願
【整理番号】　49260001
【提出日】　平成12年10月23日
【あて先】　特許庁長官殿
【国際特許分類】　G06F　9/46
【発明者】
　【住所又は居所】　東京都港区芝五丁目７番１号　日本電気株式会社内
　【氏名】　髙木　淳司
【特許出願人】
　【識別番号】　000004237
　【氏名又は名称】　日本電気株式会社
【代理人】
　【識別番号】　100088959
　【弁理士】
　【氏名又は名称】　境　廣巳
【手数料の表示】
　【予納台帳番号】　009715
　【納付金額】　21,000円
【提出物件の目録】
　【物件名】　明細書　1
　【物件名】　図面　1
　【物件名】　要約書　1
　【包括委任状番号】　9002136
【プルーフの要否】　要

【書類名】　　明細書

【発明の名称】　スタブ検索ローディングシステム及び方法、サーバ装置、クライアント装置並びにコンピュータ可読記録媒体

【特許請求の範囲】

【請求項１】　クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェースとを備えたスタブ検索ローディングシステム。

【請求項２】　クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたスタブ検索ローディングシステム。

【請求項３】　クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたスタブ検索ローディングシステム。

【請求項４】　クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

（ａ）クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

（ｂ）サーバにおいて、指定されたクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の中から１つのスタブ集合を選択するステップ、

（ｃ）サーバにおいて、指定されたスタブ名のスタブを前記選択したスタブ集合の中から選択するステップ、

（ｄ）サーバからクライアントに、前記選択されたスタブを送出するステップ、

（ｅ）クライアントにおいて、サーバから送出された前記スタブを受信するステ

ップを含むスタブ検索ローディング方法。

【請求項５】　クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

（ａ）クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

（ｂ）サーバにおいて、指定されたスタブ名とクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを生成するステップ、

（ｃ）サーバからクライアントに、前記生成したスタブを送出するステップ、

（ｄ）クライアントにおいて、サーバから送出された前記スタブを受信するステップを含むスタブ検索ローディング方法。

【請求項６】　クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

（ａ）クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

（ｂ）サーバにおいて、指定されたクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の中から１つのスタブ集合を選択し、且つ、該選択したスタブ集合の中から指定されたスタブ名のスタブを検索するステップ、

（ｃ）サーバにおいて、ステップｂの検索に失敗した場合に、指定されたスタブ名とクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを生成するステップ、

（ｄ）サーバからクライアントに、ステップｂの検索に成功した場合にはステップｂで検索されたスタブを、ステップｂの検索に失敗した場合にはステップｃで生成されたスタブを送出するステップ、

（ｅ）クライアントにおいて、サーバから送出された前記スタブを受信するステ

ップを含むスタブ検索ローディング方法。

【請求項７】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置。

【請求項８】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置。

【請求項９】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライア

ントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置。

【請求項１０】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合、及びクライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求を送出し、その応答として返されるスタブを受信するスタブ検索部を備えたクライアント装置。

【請求項１１】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部及び、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出し、その応答として返されるスタブを受信するスタブ検索部を備えたクライアント装置。

【請求項１２】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置であって、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部及び、クライアントからのスタブ要求時、指定

されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出し、その応答として返されるスタブを受信するスタブ検索部を備えたクライアント装置。

【請求項１３】　クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングプログラムを記録したコンピュータ可読記録媒体であって、

クライアントを構成するコンピュータを、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索手段として機能させ、

サーバを構成するコンピュータを、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の記憶手段、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項１４】　クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングプログラムを記録したコンピュータ可読記録媒体であって、

クライアントを構成するコンピュータを、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索手段として機能させ、

サーバを構成するコンピュータを、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるク

ライアントの種類毎に生成するスタブ生成手段、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項１５】　クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングプログラムを記録したコンピュータ可読記録媒体であって、

クライアントを構成するコンピュータを、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索手段として機能させ、

サーバを構成するコンピュータを、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の記憶手段、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成手段、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項１６】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置を構成するコンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の記憶手段、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記ス

タブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項１７】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置を構成するコンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成手段、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項１８】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをクライアントからのスタブ要求に応じて要求元のクライアントに提供するサーバ装置を構成するコンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の記憶手段、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成手段、クライアントからのスタブ名とクライアント識別子とを指定したスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェース手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項１９】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置を構成するコ

ンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合、及びクライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求を送出し、その応答として返されるスタブを受信するスタブ検索手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項２０】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置を構成するコンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部及び、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出し、その応答として返されるスタブを受信するスタブ検索手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【請求項２１】　リモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするクライアント装置を構成するコンピュータを、

クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部及び、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切な

スタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えたサーバ装置に対して、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出し、その応答として返されるスタブを受信するスタブ検索手段として機能させるプログラムを記録したコンピュータ可読記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、リモートメソッド呼び出し機構を備えたコンピュータシステムに関し、特にクライアント側で必要となるスタブをサーバ側からダウンロードして使用するコンピュータシステムにおけるスタブ検索ローディングシステム及びその方法に関する。

【０００２】

【従来の技術】

リモートメソッド呼び出し（Ｒｅｍｏｔｅ　Ｍｅｔｈｏｄ　Ｉｎｖｏｃａｔｉｏｎ；以下、ＲＭＩと称す）は、別のＪａｖａ仮想マシン上にあるＪａｖａオブジェクトのメソッドを、同じＪａｖａ仮想マシン上にあるオブジェクトのメソッドと同じように呼び出せるようにした機構であり、Ｊａｖａを使って分散システムを構築する際に使われる。

【０００３】

図１０にＲＭＩの仕組みの概要を示す。ＲＭＩは以下の手順で処理される。先ず、クライアントオブジェクトからサーバオブジェクトへの呼び出しは、クライアントに置かれたスタブに送られる((1))。スタブは、この呼び出しに対してマーシャリング処理などを行って、サーバのスケルトンに転送する((2))。スケルトンは、スタブから送られてきた呼び出しに対してアンマーシャリング処理などを行ってサーバオブジェクトへの呼び出しを生成し、サーバオブジェクトの呼び出しを行う((3))。ここで、マーシャリング処理とは、内部のデータ表現形式を

異なるシステム間で交換可能なデータ形式に変換する処理を言い、アンマーシャリング処理はその逆の処理を言う。

【０００４】

サーバオブジェクトは、呼び出されたメソッドを実行し、戻り値がある場合はそれをスケルトンに送る((4))。スケルトンは、送られた戻り値に対してマーシャリング処理などを施して、クライアントのスタブに送り返す((5))。スタブは、送られてきた戻り値に対しアンマーシャリング処理などを施し、呼び出し元のクライアントオブジェクトにメソッド呼び出しの戻り値として返す((6))。

【０００５】

このようにクライアントオブジェクトからサーバオブジェクトの呼び出しは、全てスタブを通じて行われる。つまり、スタブは、分散環境におけるクライアント側にあってサーバオブジェクトの代理人としての役割を持つ。一般には、１つのサーバオブジェクト毎に１つのスタブが必要となる。

【０００６】

スタブとスケルトンは、サーバアプリケーションの呼び出しインタフェースを記述したＩＤＬ（Interface Definition Language)ファイルをＩＤＬコンパイラでコンパイルすることで、生成することができる。スタブとスケルトンを実際に利用するときは、クライアントアプリケーション、サーバアプリケーションをコンパイルするときに、スタブ、スケルトンをリンクして一緒にコンパイルする方式（以下これを静的方式と言う）が一般的である。しかし、静的方式では、リモート手続きを呼び出すのに必要な情報はプログラムの実行時でなくコンパイル時に決定しなければならず、プログラムにリンクされたスタブと、そのプログラムの実行時に呼び出されるリモート手続きの要件との間の不一致のためにエラーが発生し、効率が低下する恐れがあった。また、全てのスタブをクライアント側で記憶するための記憶容量が必要であった。

【０００７】

そこで、このような問題点を解消するために、クライアントがスタブをサーバから動的にダウンロードして使用する方式（以下これを動的方式と言う）が提案されている（例えば特開平１０－８３３０８号公報）。以下、従来の動的方式に

ついて、図面を参照して説明する。

【０００８】

図１１に動的方式を採用した従来のコンピュータシステムの一例を示す。この従来のコンピュータシステムは、クライアントコンピュータ１０１、１０２、サーバコンピュータ１０３及びネームサーバコンピュータ１０４が、ネットワーク１０５を通じて相互に接続されている。

【０００９】

クライアントコンピュータ１０１には、Ｊａｖａ実行環境（Java Runtime Enviroment）１１０が構築されており、このＪａｖａ実行環境１１０の下で、クラス１１１が実行される。Ｊａｖａ実行環境とは、Ｊａｖａ仮想マシンを含めて、Ｊａｖａプログラムを実行させるために必要となるソフトウェアである。また、クラス１１１は、Ｊａｖａソースコードをコンパイルして作成されたクラスファイルであり、Ｊａｖａ仮想マシンが実行するコードで記述されている。図では、スタブクラスを含め、クライアントコンピュータ１０１に存在するクラスの全てをクラス１１１で表現している。

【００１０】

Ｊａｖａ実行環境１１０には、制御部１１２と、スタブ検索部１１３を有するクラスローダ１１４と、インスタンス１１５とが含まれる。スタブ検索部１１３は、サーバコンピュータ１０３からスタブクラスをダウンロードする手段である。クラスローダ１１４は、クラス１１１及びスタブ検索部１１３でダウンロードしたスタブクラスをＪａｖａ実行環境１１０にロードする手段である。インスタンス１１５は、クラスローダ１１４によってロードされたクラス１１１及びスタブクラスをインスタンス化して生成したオブジェクトである。図では、スタブクラスインスタンスを含め、Ｊａｖａ実行環境１１０に存在するインスタンスの全てをインスタンス１１５で表現している。個々のインスタンス（オブジェクト）１１５は、変数（状態）とメソッド（振る舞い）とを持つ。周知のようにオブジェクト指向プログラミングにおいては、このような複数のオブジェクト（インスタンス）が互いにメッセージを送受信し合い、その相互作用によってプログラムが実行される。制御部１１２は、Ｊａｖａ実行環境１１０の制御モジュールであ

り、Ｊａｖａ仮想マシン等を含んで構成される。制御部１１２は、クラスローダ１１４によってロードされたクラスをインスタンス化してインスタンス１１５を生成し、クライアント側のプログラムの実行を制御する。

【００１１】

他のクライアントコンピュータ１０２も、クライアントコンピュータ１０１と同様な構成を有する。

【００１２】

サーバコンピュータ１０３にも、Ｊａｖａ実行環境１２０が構築されており、このＪａｖａ実行環境１２０の下で、クラス１２１及びスケルトンクラス１２２が実行される。Ｊａｖａ実行環境１２０には、制御部１２３と、クラスローダ１２４と、インスタンス１２５とが含まれる。クラスローダ１２４は、クラス１２１及びスケルトンクラス１２２をＪａｖａ実行環境１２０にロードする手段である。インスタンス１２５は、クラスローダ１２４によってロードされたクラス１２１及びスケルトンクラス１２２をインスタンス化して生成したオブジェクトである。図では、スケルトンクラスインスタンスを含め、Ｊａｖａ実行環境１２０に存在するインスタンスの全てをインスタンス１２５で表現している。制御部１２３は、Ｊａｖａ実行環境１２０の制御モジュールであり、Ｊａｖａ仮想マシン等を含んで構成される。制御部１２３は、クラスローダ１２４によってロードされたクラス及びスケルトンクラスをインスタンス化してインスタンス１２５を生成し、サーバ側のプログラムの実行を制御する。

【００１３】

サーバコンピュータ１０３には、更に、スタブクラス１２６とスタブ検索インターフェース１２７とが設けられている。スタブクラス１２６とスケルトンクラス１２２とは、１対１に対応している。つまり、ＩＤＬファイルをコンパイルして生成されたスケルトンクラス１２２とスタブクラス１２６のペアが、サーバコンピュータ１０３で保持管理されている。そして、これらのスタブクラス１２６は、スタブクラスの名前で検索可能に保持されている。スタブ検索インターフェース１２７は、クライアントコンピュータ１０１、１０２のスタブ検索部１１３からのスタブクラス名を指定したダウンロード要求時に、指定されたスタブクラ

ス名のスタブクラス１２６を検索し、それを要求元のスタブ検索部１１３に送り返す手段である。

【００１４】

ネームサーバコンピュータ１０４は、クライアントコンピュータ１０１、１０２が、必要とするスタブクラスをどのサーバコンピュータ１０３からダウンロードして良いか分からない、つまり、必要とするスタブクラスを保持するサーバコンピュータ１０３をクライアントコンピュータ１０１、１０２が知らない場合に、該当するサーバコンピュータ１０３をクライアントコンピュータ１０１、１０２に教える機能を持つコンピュータである。典型的には、スタブクラス名毎に、そのスタブクラスを保持するサーバコンピュータ１０３の識別子を保持するテーブルを保持管理しており、クライアントコンピュータ１０１、１０２からスタブクラス名を指定した問い合わせがあった際に、そのテーブルを参照して、該当するサーバコンピュータ１０３の識別子を返すように構成される。

【００１５】

次に従来のコンピュータシステムにおけるＲＭＩの処理手順を、図１２のフローチャートを参照して説明する。

【００１６】

クライアントコンピュータ１０１のＪａｖａ実行環境１１０の制御部１１２は、或るクラス１１１のインスタンス１１５から、サーバコンピュータ１０３における或るクラス１２１の或るメソッドを呼び出すリモートメソッド呼び出しが行われると、その呼び出しに使うスタブクラス１２６のインスタンス１１５がＪａｖａ実行環境１１０に存在するか否かを判別する（Ｓ１０１）。存在した場合には、図１０で説明したような手順に従って、そのスタブクラス１２６のインスタンス１１５を使って当該リモートメソッド呼び出しを実現する（Ｓ１０２）。

【００１７】

今回の呼び出しに使うスタブクラス１２６のインスタンス１１５がＪａｖａ実行環境１１０に存在しなかった場合、制御部１１２は、今回の呼び出しに使うスタブクラス１２６がクライアントコンピュータ１０１のクラス１１１中に存在するか否かを判別する（Ｓ１０３）。存在した場合、制御部１１２はクラスローダ

１１４によって、今回の呼び出しに使うスタブクラスのクラス１１１をＪａｖａ実行環境１１０にロードし（Ｓ１０４）、それをインスタンス化して、今回の呼び出しに使うスタブクラス１２６のインスタンス１１５を生成する（Ｓ１０５）。そして、そのスタブクラス１２６のインスタンス１１５を使って当該リモートメソッド呼び出しを実現する（Ｓ１０２）。

【００１８】

今回の呼び出しに使うスタブクラス１２６がクライアントコンピュータ１０１のクラス１１１中に存在しなかった場合、今回の呼び出しに使うスタブクラス１２６の存在場所を、クライアントコンピュータ１０１側で知っているか否かを判別する（Ｓ１０６）。今回の呼び出しに使うスタブクラス１２６の存在場所は、例えばその呼び出し中に含めることができ、またクラスローダ１１４で各スタブクラス１２６の所在を管理することもできるため、そのような場合にはクライアントコンピュータ１０１側で知っていることになる。他方、クライアントコンピュータ１０１側が今回の呼び出しに使うスタブクラス１２６の存在場所を知らなかった場合、制御部１１２は、スタブクラスのクラス名を指定してネームサーバコンピュータ１０４に問い合わせる（Ｓ１０７）。これに応答してネームサーバコンピュータ１０４は、指定されたクラス名のスタブクラスの存在場所を通知する（Ｓ１０８）。

【００１９】

今回の呼び出しに使うスタブクラス１２６の存在場所が明らかになると、制御部１１２は、スタブ検索部１１３によって、前記存在場所で一意に定まるサーバコンピュータ１０３に対し、クラス名を指定してスタブクラスを要求する（Ｓ１０９）。サーバコンピュータ１０３のスタブ検索インターフェース１２７は、指定されたクラス名のスタブクラスをスタブクラス１２６から検索し、要求元のクライアントコンピュータ１０１に返す（Ｓ１１０、Ｓ１１１）。この返されてきたスタブクラスは、クラスローダ１１４によってＪａｖａ実行環境１１０にロードされ（Ｓ１０４）、制御部１１２によってそのインスタンスが生成される（Ｓ１０５）。そして、そのスタブクラス１２６のインスタンス１１５を使って当該リモートメソッド呼び出しを実現する（Ｓ１０２）。なお、スタブ検索インター

フェース１２７による検索で該当するスタブクラスが見つからなかった場合、その旨がクライアントコンピュータ１０１に返される（Ｓ１１２）。この場合、クライアントコンピュータ１０１側ではスタブクラスのダウンロード要求は失敗に終わり、例外処理によりリモートメソッド呼び出しはエラー終了する。

【００２０】

【発明が解決しようとする課題】

スタブクラスはマーシャリング処理等を行うスタブクラスインスタンスのクラスであり、実行されるクライアントコンピュータのマシンやＯＳやＪａｖａ実行環境などで特定される、スタブクラスの実行環境に依存した構成となる。このため、サーバコンピュータ１０３のＪａｖａ実行環境１２０の下で実行されるインスタンス１２５のメソッドを呼び出すのに必要なクライアント側のスタブクラスは、実行されるＪａｖａ実行環境１１０の種類が違えば、そのクラス名は同じであっても、中身は異なる。

【００２１】

図１１で説明した従来のコンピュータシステムでは、クライアントコンピュータ１０１のスタブ検索部１１３はクラス名を指定してスタブクラスを要求し、サーバコンピュータ１０３のスタブ検索インターフェース１２７は指定されたクラス名を持つスタブクラスを返すようにしている。従って、クライアントコンピュータ１０１とクライアントコンピュータ１０２のＪａｖａ実行環境１１０の種類が同じであれば問題はないが、相違すれば、サーバコンピュータ１０３からダウンロードされるスタブクラスに適合するＪａｖａ実行環境を備えたクライアントコンピュータだけが正常に動作し、他のクライアントコンピュータは正常な動作を保証できない。これは、Ｊａｖａ実行環境の種類が異なるために同じスタブクラスを使用できない複数のクライアントコンピュータ１０１、１０２から同じサーバコンピュータ１０３に対して、動的方式によってスタブを獲得してリモートメソッド呼び出しを行うことができないことを意味する。また、Ｊａｖａ実行環境の種類が異なるために同じスタブクラスを使用できない複数のクライアントコンピュータからリモートメソッド呼び出しを行うことができるようにするには、複数のサーバコンピュータを用意しなければならないことを意味する。

【００２２】

本発明の目的は、１つのサーバコンピュータから、Ｊａｖａ実行環境の種類が異なる複数のクライアントコンピュータに対して、各クライアントコンピュータにおけるリモートメソッド呼び出しに使えるスタブクラスをダウンロードできるようにすることにある。

【００２３】

本発明の別の目的は、クライアントコンピュータから要求されたスタブクラスがサーバコンピュータに存在しない場合、適切なスタブクラスを動的に生成してダウンロードできるようにすることにある。

【００２４】

【課題を解決するための手段】

本発明の第１のスタブ検索ローディングシステムは、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返すスタブ検索インターフェースとを備えている。

【００２５】

本発明の第２のスタブ検索ローディングシステムは、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えている。

【００２６】

また本発明の第３のスタブ検索ローディングシステムは、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディングシステムであって、

クライアント側に、スタブ名とクライアント識別子とを指定したスタブ要求をサーバに送出して、その応答として返されるスタブを受信するスタブ検索部を備え、

サーバ側に、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合と、前記スケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部と、クライアントからのスタブ要求時、指定されたスタブ名とクライアント識別子に基づいて前記スタブ集合の中から適切なスタブを検索して要求元のクライアントに返し、適切なスタブが前記スタブ集合に存在しなければ指定されたスタブ名とクライアント識別子に基づいて前記スタブ生成部に要求元のクライアントの実行環境に適したスタブを生成させ、生成されたスタブを要求元のクライアントに返すスタブ検索インターフェースとを備えている。

【００２７】

また本発明の第１のスタブ検索ローディング方法は、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

（ａ）クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

（ｂ）サーバにおいて、指定されたクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の中から１つのスタブ集合を選択するステップ、

（ｃ）サーバにおいて、指定されたスタブ名のスタブを前記選択したスタブ集合の中から選択するステップ、

（ｄ）サーバからクライアントに、前記選択されたスタブを送出するステップ、

（ｅ）クライアントにおいて、サーバから送出された前記スタブを受信するステップを含んでいる。

【００２８】

本発明の第２のスタブ検索ローディング方法は、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

（ａ）クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

（ｂ）サーバにおいて、指定されたスタブ名とクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを生成するステップ、

（ｃ）サーバからクライアントに、前記生成したスタブを送出するステップ、

（ｄ）クライアントにおいて、サーバから送出された前記スタブを受信するステップを含んでいる。

【００２９】

本発明の第３のスタブ検索ローディング方法は、クライアントからサーバにリモートメソッド呼び出しを行う際にクライアント側で必要となるスタブをサーバ側からダウンロードするスタブ検索ローディング方法であって、

（ａ）クライアントからサーバに、スタブ名とクライアント識別子とを指定したスタブ要求を送出するステップ、

（ｂ）サーバにおいて、指定されたクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使

うスタブを実行環境の異なるクライアントの種類毎に用意したスタブ集合の中から１つのスタブ集合を選択し、且つ、該選択したスタブ集合の中から指定されたスタブ名のスタブを検索するステップ、

（ｃ）サーバにおいて、ステップｂの検索に失敗した場合に、指定されたスタブ名とクライアント識別子に基づいて、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを生成するステップ、

（ｄ）サーバからクライアントに、ステップｂの検索に成功した場合にはステップｂで検索されたスタブを、ステップｂの検索に失敗した場合にはステップｃで生成されたスタブを送出するステップ、

（ｅ）クライアントにおいて、サーバから送出された前記スタブを受信するステップを含んでいる。

【００３０】

ここで、クライアント識別子は、スタブが実行されるクライアントの実行環境をサーバ側で特定できるものであれば、どのようなものでも良い。例えば、クライアントコンピュータの機種名、型番、ＯＳの種類、Ｊａｖａ実行環境の種類、ネットワークアドレスや、これらに限定されずなにかしらのクライアントの情報、またはこれらの二つ以上を組み合わせたものでも良い。また、クライアントから明示的に送信される情報であっても良いし、ネットワークアドレスのようにスタブクラスの要求を送信することによって暗黙的に送信されるものであっても良い。

【００３１】

【作用】

本発明の第１のスタブ検索ローディングシステム及びその方法にあっては、スタブを実行環境の異なるクライアントの種類毎にサーバ側に用意しておき、クライアントからサーバへのスタブ要求時にスタブ名に加えてそのクライアントの実行環境を特定し得るクライアント識別子を通知し、サーバ側では、指定されたスタブ名とクライアント識別子に基づいて適切なスタブを検索して要求元のクライアントに返すようにしたので、１つのサーバコンピュータから、実行環境の種類が異なる複数のクライアントコンピュータに対して適切なスタブをダウンロード

することが可能となる。

【００３２】

本発明の第２のスタブ検索ローディングシステム及びその方法にあっては、クライアントからのリモートメソッド呼び出し時にサーバ側で使うスケルトンと組にして使うスタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部をサーバ側に用意しておき、クライアントからサーバへのスタブ要求時にスタブ名に加えてそのクライアントの実行環境を特定し得るクライアント識別子を通知し、サーバ側では、指定されたクライアント識別子のクライアントの実行環境に適したスタブを生成して要求元のクライアントに返すようにしたので、１つのサーバコンピュータから、実行環境の種類が異なる複数のクライアントコンピュータに対して適切なスタブをダウンロードすることが可能となる。

【００３３】

本発明の第３のスタブ検索ローディングシステム及びその方法にあっては、スタブを実行環境の異なるクライアントの種類毎にサーバ側に用意しておくと共に、スタブを実行環境の異なるクライアントの種類毎に生成するスタブ生成部をサーバ側に用意しておき、クライアントからサーバへのスタブ要求時にスタブ名に加えてそのクライアントの実行環境を特定し得るクライアント識別子を通知し、サーバ側では、指定されたスタブ名とクライアント識別子に基づいて適切なスタブを検索して要求元のクライアントに返すか、若し適切なスタブが存在しなければ、指定されたクライアント識別子のクライアントの実行環境に適したスタブを生成して要求元のクライアントに返すようにしたので、１つのサーバコンピュータから、実行環境の種類が異なる複数のクライアントコンピュータに対して適切なスタブをダウンロードすることが可能となる。

【００３４】

【発明の実施の形態】

次に本発明の実施の形態の例について図面を参照して詳細に説明する。

【００３５】

図１を参照すると、本発明の一実施の形態にかかるコンピュータシステムは、クライアントコンピュータ１～３、サーバコンピュータ４及びネームサーバコン

ピュータ５が、ネットワーク６を通じて相互に接続されている。ネットワーク６は、ＬＡＮ、ＷＡＮ、公衆電話網、インターネットなど、コンピュータ間でデジタルデータを送受信できる任意のネットワークであり、その通信媒体も有線、無線或いはそれ以外のどのような媒体であっても構わない。

【００３６】

クライアントコンピュータ１には、Ｊａｖａ実行環境(1) １０が構築されており、このＪａｖａ実行環境１０の下で、クラス１１が実行される。クラス１１は、Ｊａｖａソースコードをコンパイルして作成されたクラスファイルであり、Ｊａｖａ仮想マシンが実行するコードで記述されている。図では、スタブクラスを含め、クライアントコンピュータ１に存在するクラスの全てをクラス１１で表現している。

【００３７】

Ｊａｖａ実行環境１０には、制御部１２と、スタブ検索部１３を有するクラスローダ１４と、インスタンス１５とが含まれる。スタブ検索部１３は、サーバコンピュータ４からスタブクラスをダウンロードする手段である。クラスローダ１４は、クラス１１及びスタブ検索部１３でダウンロードしたスタブクラスをＪａｖａ実行環境１０にロードする手段である。インスタンス１５は、クラスローダ１４によってロードされたクラス１１及びスタブクラスをインスタンス化して生成したオブジェクトである。図では、スタブクラスインスタンスを含め、Ｊａｖａ実行環境１０に存在するインスタンスの全てをインスタンス１５で表現している。制御部１２は、Ｊａｖａ実行環境１０の制御モジュールであり、Ｊａｖａ仮想マシン等を含んで構成される。制御部１２は、クラスローダ１４によってロードされたクラスをインスタンス化してインスタンス１５を生成し、クライアント側のプログラムの実行を制御する。

【００３８】

他のクライアントコンピュータ２，３も、クライアントコンピュータ１と同様な構成を有するが、コンピュータのマシン、ＯＳなどが各クライアントコンピュータ１、２、３で異なるため、構築されているＪａｖａ実行環境の種類が相違している。つまり、本例の場合、Ｊａｖａ実行環境(1) 、Ｊａｖａ実行環境(2) 、

Ｊａｖａ実行環境(3) の３種類のＪａｖａ実行環境があり、クライアントコンピュータ１にはＪａｖａ実行環境(1) １０が、クライアントコンピュータ２にはＪａｖａ実行環境(2) １６が、クライアントコンピュータ３にはＪａｖａ実行環境(3) １７が、それぞれ構築されている。

【００３９】

サーバコンピュータ４には、Ｊａｖａ実行環境(1) ２０が構築されており、このＪａｖａ実行環境２０の下で、クラス２１及びスケルトンクラス２２が実行される。Ｊａｖａ実行環境２０には、制御部２３と、クラスローダ２４と、インスタンス２５とが含まれる。クラスローダ２４は、クラス２１及びスケルトンクラス２２をＪａｖａ実行環境２０にロードする手段である。インスタンス２５は、クラスローダ２４によってロードされたクラス２１及びスケルトンクラス２２をインスタンス化して生成したオブジェクトである。図では、スケルトンクラスインスタンスを含め、Ｊａｖａ実行環境２０に存在するインスタンスを全てインスタンス２５で表現している。制御部２３は、Ｊａｖａ実行環境２０の制御モジュールであり、Ｊａｖａ仮想マシン等を含んで構成される。制御部２３は、クラスローダ２４によってロードされたクラス及びスケルトンクラスをインスタンス化してインスタンス２５を生成し、サーバ側のプログラムの実行を制御する。

【００４０】

サーバコンピュータ４には、更に、スタブクラス２６－１～２６－３の記憶部２９と、クライアント識別部２８を有するスタブ検索インターフェース２７とが設けられている。スタブクラス２６－１は、Ｊａｖａ実行環境(1) を有するクライアントコンピュータ１においてリモートメソッド呼び出しを行う際に利用可能なスタブクラスを集めたものである。同様に、スタブクラス２６－２、２６－３は、Ｊａｖａ実行環境(2) 、(3) を有するクライアントコンピュータ２、３においてリモートメソッド呼び出しを行う際に利用可能なスタブクラスを集めたものである。各スタブクラス２６－１～２６－３中の個々のスタブクラスは、スタブクラスの名前で検索可能に保持されている。また、同じリモートメソッド呼び出しに使うスタブクラス、つまり、サーバコンピュータ４における同じサーバオブジェクトのメソッドを呼び出すのに使うスタブクラスには、スタブクラス２６－

１～２６－３で同じクラス名が付与されている。

【００４１】

図２に、サーバコンピュータ４に保持するスタブクラス２６－１～２６－３を生成する構成例を示す。図２において、５０－１～５０－ｎは、サーバコンピュータ４が提供するサービスのインターフェースをＪａｖａのインターフェースクラス（メソッドの名前と型の情報だけを持ち、メソッドの実装コードを持たないクラス）で定義したもので、サーバコンピュータ４が提供するサービス毎に存在する。

【００４２】

ＲＭＩコンパイラ５１－１は、Ｊａｖａのインターフェースクラスから Ｊａｖａ実行環境(1) 用のスケルトンクラス及びスタブクラスを生成するコンパイラであり、このＲＭＩコンパイラ５１－１にインターフェースクラス５０－１～５０－ｎのそれぞれをコンパイルさせると、図１のサーバコンピュータ４に保持すべきスケルトンクラス２２及びスタブクラス２６－１が生成される。また、ＲＭＩコンパイラ５１－２は、Ｊａｖａのインターフェースクラスから Ｊａｖａ実行環境(2) 用のスケルトンクラス及びスタブクラスを生成するコンパイラであり、このＲＭＩコンパイラ５１－２にインターフェースクラス５０－１～５０－ｎのそれぞれをコンパイルさせると、図１のサーバコンピュータ４に保持すべきスタブクラス２６－２が生成される。同時に生成されるスケルトンクラス５２は使用しない。更に、ＲＭＩコンパイラ５１－３は、Ｊａｖａのインターフェースクラスから Ｊａｖａ実行環境(3) 用のスケルトンクラス及びスタブクラスを生成するコンパイラであり、このＲＭＩコンパイラ５１－３にインターフェースクラス５０－１～５０－ｎのそれぞれをコンパイルさせると、図１のサーバコンピュータ４に保持すべきスタブクラス２６－３が生成される。同時に生成されるスケルトンクラス５３は使用しない。ＲＭＩコンパイラ５１－１～５１－３としては、例えばｒｍｉｃを使用することができる。

【００４３】

図２の構成例では、１つのＲＭＩコンパイラで１つのＪａｖａ実行環境用のスタブクラスを生成したが、ＲＭＩコンパイラの種類によっては、作成するスタブ

クラスの種類をコンパイラオプションで指定できるものがあり、そのようなＲＭＩコンパイラを使えば、１つのＲＭＩコンパイラで複数のＪａｖａ実行環境用のスタブクラスを生成することができる。例えばｒｍｉｃでは、コンパイラオプションの指定により、バージョン1.1 のＪＲＭＰスタブプロトコルバージョンのスタブとスケルトン、バージョン1.2 のＪＲＭＰスタブプロトコルバージョンのスタブとスケルトン、バージョン1.1 、1.2 の両方のＪＲＭＰスタブプロトコルバージョンと互換性のあるスタブとスケルトンを作成することができる。図３に、そのような機能を持つＲＭＩコンパイラ５４を使って複数のＪａｖａ実行環境用のスタブクラスを生成する構成例を示す。

【００４４】

図３において、ＲＭＩコンパイラ５４に、Ｊａｖａ実行環境(1) 用のオプション指定（例えばバージョン1.1 、1.2 の両方のＪＲＭＰスタブプロトコルバージョンと互換性のあるスタブとスケルトンを作成する指定）を与えて、インターフェースクラス５０－１～５０－ｎのそれぞれをコンパイルさせると、図１のサーバコンピュータ４に保持すべきスケルトンクラス２２及びスタブクラス２６－１が生成される。また、ＲＭＩコンパイラ５４に、Ｊａｖａ実行環境(2) 用のオプション指定（例えばバージョン1.1 のＪＲＭＰスタブプロトコルバージョンのスタブとスケルトンを作成する指定）を与えて、インターフェースクラス５０－１～５０－ｎのそれぞれをコンパイルさせると、図１のサーバコンピュータ４に保持すべきスタブクラス２６－２が生成される。同時に生成されるスケルトンクラス５２は使用しない。更に、ＲＭＩコンパイラ５４に、Ｊａｖａ実行環境(3) 用のオプション指定（例えばバージョン1.2 のＪＲＭＰスタブプロトコルバージョンのスタブとスケルトンを作成する指定）を与えて、インターフェースクラス５０－１～５０－ｎのそれぞれをコンパイルさせると、図１のサーバコンピュータ４に保持すべきスタブクラス２６－３が生成される。同時に生成されるスケルトンクラス５３は使用しない。

【００４５】

再び図１を参照すると、クライアント識別部２８は、クライアントコンピュータ１～３からのスタブクラス名を指定したスタブ要求時に、同時に送られてくる

クライアント識別子を識別し、スタブクラス２６－１～２６－３の内、どのスタブクラスの集合を検索対象とすべきかを決定する部分である。クライアント識別部２８は、例えば、クライアント識別子とスタブクラスの集合名（スタブクラス２６－１～２６－３の名前）との対応を保持するテーブル（図示せず）を保持しており、このテーブルをクライアント識別子で検索して該当するスタブクラスの集合を特定する。スタブ検索インターフェース２７は、クライアント識別部２８で決定された何れか１つのスタブクラス２６－１～２６－３から、スタブ要求で指定されたスタブクラス名を持つスタブクラスを検索し、それを要求元のクライアントコンピュータ１～３に送り返す手段である。

【００４６】

ネームサーバコンピュータ５は、クライアントコンピュータ１～３からのスタブクラス名を指定した問い合わせに応答して、例えばスタブクラス名毎にそのスタブクラスの所在場所情報を保持管理するテーブルを参照して、そのスタブクラスの所在場所を見つけ、それを問い合わせ元のクライアントコンピュータ１～３に返す機能を持つコンピュータである。

【００４７】

次に、図１に示したコンピュータシステムにおけるＲＭＩの処理手順を、本発明の特徴部分であるスタブクラスのダウンロード動作を中心に図４のフローチャートを参照して説明する。

【００４８】

クライアントコンピュータ１のＪａｖａ実行環境１０の制御部１２は、例えば或るクラス１１のインスタンス１５から、サーバコンピュータ４における或るクラス２１の或るメソッドを呼び出すリモートメソッド呼び出しが行われると、その呼び出しに使うスタブクラスのインスタンスがＪａｖａ実行環境１０のインスタンス１５中に存在するか否かを判別する（Ｓ１）。存在した場合には、図１０で説明したような手順に従って、そのスタブクラスのインスタンス１１５を使って当該リモートメソッド呼び出しを実現する（Ｓ２）。

【００４９】

今回の呼び出しに使うスタブクラスのインスタンス１５がＪａｖａ実行環境１

０に存在しなかった場合、制御部１２は、今回の呼び出しに使うスタブクラスがクライアントコンピュータ１のクラス１１中に存在するか否かを判別する（Ｓ３）。存在した場合、制御部１２はクラスローダ１４によって、今回の呼び出しに使うスタブクラスのクラス１１をＪａｖａ実行環境１０にロードし（Ｓ４）、それをインスタンス化して、今回の呼び出しに使うスタブクラスのインスタンス１５を生成する（Ｓ５）。そして、そのスタブクラスのインスタンス１５を使って当該リモートメソッド呼び出しを実現する（Ｓ２）。

【００５０】

今回の呼び出しに使うスタブクラスがクライアントコンピュータ１のクラス１１中に存在しなかった場合、今回の呼び出しに使うスタブクラスの存在場所を、クライアントコンピュータ１側で知っているか否かを従来例で説明したと同様に判別する（Ｓ６）。知っている場合にはステップＳ９へ進み、知らなかった場合は、スタブクラスのクラス名を指定してネームサーバコンピュータ５に問い合わせる（Ｓ７）。これに応答してネームサーバコンピュータ５は、指定されたクラス名のスタブクラスの存在場所を通知する（Ｓ８）。

【００５１】

今回の呼び出しに使うスタブクラスの存在場所が明らかになると、制御部１２は、スタブ検索部１３によって、前記存在場所で一意に定まるサーバコンピュータ４に対し、クラス名及びクライアント識別子を指定してスタブクラスを要求する（Ｓ９）。ここで、クライアント識別子は、クライアントコンピュータ１の機種または型番またはＯＳの種類またはＪａｖａ実行環境(1) の種類または制御部１２の種類またはネットワークアドレスまたはこれらに限定されずなにかしらのクライアントの情報、またはこれらの二つ以上を組み合わせたもので、スタブ検索部１３によって明示的に送信される情報であっても、ネットワークアドレスのようにスタブクラスの要求を送信することによって暗黙的に送信されてサーバコンピュータ４のクライアント識別部２８がその情報によってクライアントコンピュータを識別することができる情報であってもよい。要するに、クライアント識別子は、ダウンロードしたスタブが実行される実行環境を直接的または間接的に特定できるものであれば良い。

【００５２】

サーバコンピュータ４のスタブ検索インターフェース２７は、クライアント識別部２８によって、受信したクライアント識別子に基づいてスタブクラス２６－１～２６－３の内、どのスタブクラスの集合を検索対象とすべきかを決定し、この決定した何れか１つのスタブクラス２６－１～２６－３から、スタブ要求で指定されたスタブクラス名を持つスタブクラスを検索する（Ｓ１０）。今の場合は、Ｊａｖａ実行環境(1) 用のスタブクラス２６－１からスタブクラスが検索される。そして、該当するスタブクラスが存在した場合には、それを要求元のクライアントコンピュータ１に送り返す（Ｓ１１）。また、存在しなかった場合には、クライアントコンピュータ１に対して該当するスタブクラスを保持していない旨を通知する（Ｓ１２）。この場合、スタブクラスのダウンロード要求は失敗に終わり、クライアントコンピュータ１では例外処理が実施され、例えばリモートメソッド呼び出しがエラー終了する。

【００５３】

クライアントコンピュータ１のスタブ検索部１３は、要求先のサーバコンピュータ４からスタブクラスが送られてくると、これをクラスローダ１４に渡し、クラスローダ１４はＪａｖａ実行環境１０にロードする（Ｓ４）。そして、制御部１２はこのロードされたスタブクラスをインスタンス化してインスタンス１５を生成し（Ｓ５）、インスタンス１５を使って当該リモートメソッド呼び出しを実現する（Ｓ２）。

【００５４】

以上は、クライアントコンピュータ１を例にして動作を説明したが、他のクライアントコンピュータ２、３におけるＲＭＩの処理も同様に行われる。但し、クライアントコンピュータ２、３におけるサーバコンピュータ４へのスタブクラスの要求時には、クライアントコンピュータ２、３のクライアント識別子が送られるので、サーバコンピュータ４のスタブ検索インターネット２７は、クライアントコンピュータ２からのスタブクラス要求時にはＪａｖａ実行環境(2) 用のスタブクラス２６－２を検索し、クライアントコンピュータ３からのスタブクラス要求時にはＪａｖａ実行環境(3) 用のスタブクラス２６－３を検索する。

【００５５】

このようにして本実施の形態にかかるコンピュータシステムは、１つのサーバコンピュータ４から、Ｊａｖａ実行環境の種類が異なる複数のクライアントコンピュータ１～３に対して、各クライアントコンピュータ１～３におけるリモートメソッド呼び出しに使えるスタブクラスをダウンロードし、同じサーバオブジェクトのメソッドに対するリモートメソッド呼び出しを可能にしている。このため、構成が様々に異なるクライアントコンピュータ１～３に対して同じ入口のサービスを一つだけサーバコンピュータに用意しておくだけで済ますことができる。

【００５６】

次に、具体的な実施例を用いて本実施の形態の動作を補足説明する。クライアントコンピュータ１上のクライアントFoo がサーバにリモートメソッド呼び出しを行うにあたってBankService インターフェースを実装したクラスであるBankServiceImpl クラスのスタブクラスが必要になった時、クライアントコンピュータ１はネームサーバコンピュータ５にBankServiceImpl のスタブクラスがどこにあるかを問い合わせると、 http://japan/BankServiceImpl Stub.classと返って来た。

【００５７】

そこで、クライアントコンピュータ１は、japan サーバに対して、HTTPプロトコルを使用して、HTTPヘッダにクライアント識別子として "User-Agent: Foo"という文字列を付加して http://japan/BankServiceImpl Stub.classを要求する。

【００５８】

サーバコンピュータ４は、Foo という種類のクライアント識別子にマッチするものとして http://japan/client/Foo/BankServerImpl Stub.classを取り出してクライアントコンピュータ１に返す。

【００５９】

一方、クライアントコンピュータ２上のクライアントBar がサーバにリモートメソッド呼び出しを行う際は、ネームサーバコンピュータに問い合わせて http://japan/BankServiceImpl Stub.classとかえって来る所までは同様である。次に、japan サーバに対して、 HTTP プロトコルを使用して japanサーバに http://

japan/BankServiceImpl Stub.classを要求する際、HTTPヘッダにクライアント識別子として "User-Agent: Bar"という文字列を付加する。サーバコンピュータ４は、今度は Barという種類のクライアント識別子にマッチするものとして http://japan/client/Bar/BankServerImpl Stub.classを取り出してクライアントコンピュータ２に返す。

【００６０】

クライアントコンピュータ１及び２にダウンロードされインスタンス化されたスタブクラスは、いずれも japanサーバの BankServiceImplのスケルトンと通信し、リモートメソッド呼び出しを実現する。

【００６１】

【発明の他の実施の形態】

図５は本発明の別の実施の形態にかかるコンピュータシステムのブロック図であり、図１と同一符号は同一部分を示し、３０はスタブ生成部である。本実施の形態が図１に示した実施の形態と相違するところは、サーバコンピュータ４が、実行環境の異なるクライアントの種類毎のスタブを生成するスタブ生成部３０を備え、スタブクラスを要求したクライアントコンピュータ１～３に返却すべき適切なスタブクラスが記憶部２９内のスタブクラス２６－１～２６－３中に存在しなかった場合、スタブ検索インターフェース２７がスタブクラス名とクライアント識別子とを指定したスタブ生成要求をスタブ生成部３０に出し、スタブ生成部３０が該当するスタブを動的に生成し、スタブ検索インターフェース２７がその生成されたスタブを要求元のクライアントコンピュータ１～３に返すようにした点にある。また、新たに生成されたスタブクラスを後のスタブ要求に備えて記憶部２９に保存するようにした点にある。

【００６２】

図６にスタブ生成部３０の構成例を示す。図６において、６０－１～６０－ｍは、サーバコンピュータ４が提供するサービスのインターフェースをＪａｖａのインターフェースクラス（メソッドの名前と型の情報だけを持ち、メソッドの実装コードを持たないクラス）で定義したもので、サーバコンピュータ４が提供するサービス毎に存在する。６１－１～６１－ｎは、図２及び図３で説明したもの

と同様なＲＭＩコンパイラであり、それぞれ、Ｊａｖａのインターフェースクラスからスケルトンクラス及びスタブクラスを生成する。６２は、スタブ検索インターフェース２７から、スタブクラス名とクライアント識別子とを指定したスタブ生成要求を受け取って各部を制御する制御部である。

【００６３】

制御部６２は、各インターフェースクラス６０－１～６０－ｍがどのスタブクラス名に対応するものであるかを内部のテーブル（図示せず）によって管理すると共に、クライアント識別子を識別して、そのクライアント識別子のクライアントの実行環境で実行するスタブを生成するには複数存在するＲＭＩコンパイラ６１－１～６１－ｎのうち、どれを使用すべきであるか、またどのようなコンパイラオプションを指定すべきであるかを、例えばクライアント識別子とＲＭＩコンパイラ６１－１～６１－ｎとの対応表（図示せず）によって管理している。そして制御部６２は、スタブ検索インターフェース２７からのスタブ生成要求で指定されたスタブクラス名によって選択したインターフェースクラス６０－１～６０－ｍの１つを、同じくスタブ生成要求で指定されたクライアント識別子によって選択したＲＭＩコンパイラ６１－１～６１－ｎの１つに入力して（必要ならばコンパイラオプションも指定する）、コンパイルを行わせ、そのＲＭＩコンパイラで生成されたスケルトンクラスとスタブクラスのうち、スタブクラスを要求元のスタブ検索インターフェース２７に返却する処理を行う。なお、制御部６２は、該当するインターフェースクラスが存在しないか、該当するＲＭＩコンパイラが存在しないため、該当するスタブを生成することができない場合には、その旨をスタブ検索インターフェース２７に通知する。

【００６４】

次に、本実施の形態にかかるコンピュータシステムにおけるＲＭＩの処理手順を、図１の実施の形態との相違点を中心に説明する。

【００６５】

図７は本実施の形態にかかるコンピュータシステムにおけるＲＭＩの処理手順の一例を示すフローチャートであり、図４と同一符号は同一ステップを示す。図４と比較すると分かるように、本実施の形態では、スタブ検索インターフェース

２７によるスタブクラスの検索で該当するスタブクラスがスタブクラス２６－１～２６－３中に存在しないと判断された場合（Ｓ１０でＮＯ）、スタブ検索インターフェース２７からスタブクラス名とクライアント識別子とを通知されてスタブ生成部３０が起動され、スタブ生成部３０の制御部６２が、インターフェースクラス６０－１～６０－ｍ中にスタブクラス名に対応するインターフェースクラスが存在するか否か、ＲＭＩコンパイラ６１－１～６１－ｎ中にクライアント識別子に対応するＲＭＩコンパイラが存在するか否かを判定することによって、要求されたスタブクラスを生成できるか否かを調べる（Ｓ２１）。そして、生成可能な場合、スタブ生成部３０の制御部６２の制御の下に該当するスタブクラスの動的な生成が行われ、生成されたスタブクラスがスタブ検索インターフェース２７に戻される（Ｓ２２）。スタブ検索インターフェース２７はそれを要求元のクライアントコンピュータ１～３に返すと共に記憶部２９に保存する（Ｓ１１）。他方、スタブ生成部３０で該当するスタブクラスが生成できない場合、その旨がスタブ生成部３０からスタブ検索インターフェース２７に返され、スタブ検索インターフェース２７は要求元のクライアントコンピュータ１～３に該当するスタブクラスが存在しない旨を通知する（Ｓ１２）。

【００６６】

このようにして本実施の形態にかかるコンピュータシステムは、１つのサーバコンピュータ４から、Ｊａｖａ実行環境の種類が異なる複数のクライアントコンピュータ１～３に対して、各クライアントコンピュータ１～３におけるリモートメソッド呼び出しに使えるスタブクラスを必要に応じて動的に生成してダウンロードし、同じサーバオブジェクトのメソッドに対するリモートメソッド呼び出しを可能にしている。このため、構成が様々に異なるクライアントコンピュータ１～３に対して同じ入口のサービスを一つだけサーバコンピュータに用意しておくだけで済ますことができ、また、予め全ての構成に対応するスタブクラスをサーバコンピュータ４に用意しておく必要がなくなる。

【００６７】

なお、本実施の形態では、記憶部２９にスタブクラスが１つも記憶されない状態でサーバコンピュータ４を立ち上げるようにしても良いし、サーバコンピュー

タ４の立ち上げ時点で図１の実施の形態と同様に記憶部２９に幾つかのスタブクラスを予め記憶させておくようにしても良い。また、スタブ生成部３０で生成されたスタブクラスを記憶部２９に保存するようにしたが、保存しないようにしても良い。

【００６８】

【発明の他の実施の形態】

図８は本発明の更に別の実施の形態にかかるコンピュータシステムのブロック図であり、図５と同一符号は同一部分を示す。本実施の形態が図５に示した実施の形態と相違するところは、サーバコンピュータ４が、スタブクラスの集合を一切保持せずに、クライアントコンピュータ１～３からダウンロード要求されたスタブクラスの全てを動的に生成するようにした点にある。そのため、図５のスタブクラス２６－１～２６－３の記憶部２９は省略されている。

【００６９】

スタブ検索インターフェース２７は、クライアントコンピュータ１～３からスタブクラス名及びクライアント識別子を指定したスタブ要求を受けた場合、スタブ生成部３０に対してスタブクラス名とクライアント識別子を通知してスタブクラスの生成を指示する。スタブ生成部３０は図５の実施の形態で説明したものと同様の方法でスタブクラスを生成してスタブ検索インターフェース２７に返し、スタブ検索インターフェース２７はそれを要求元のクライアントコンピュータ１～３に送る。スタブ生成部３０が該当するスタブクラスを生成できない場合、スタブ検索インターフェース２７はスタブクラスが見つからない旨を要求元のクライアントコンピュータ１～３に通知する。

【００７０】

図９は本実施の形態にかかるコンピュータシステムにおけるＲＭＩの処理手順の一例を示すフローチャートであり、図７と同一符号は同一ステップを示す。図７と比較すると分かるように、本実施の形態では、クライアントコンピュータからスタブ要求を受けた場合、スタブ検索インターフェース２７からスタブ生成部３０が直ちに起動され、スタブ生成部３０は、スタブクラスを生成できるか否かを調べる（Ｓ２１）。生成可能な場合、スタブ生成部３０は該当するスタブクラ

スを生成してスタブ検索インターフェース２７に渡し（Ｓ２２）、スタブ検索インターフェース２７はそれを要求元のクライアントコンピュータ１～３に返す（Ｓ１１）。他方、該当するスタブクラスが生成できない場合、その旨がスタブ生成部３０からスタブ検索インターフェース２７に通知され、スタブ検索インターフェース２７は要求元のクライアントコンピュータ１～３に該当するスタブクラスが存在しない旨を通知する（Ｓ１２）。

【００７１】

このようにして本実施の形態にかかるコンピュータシステムは、１つのサーバコンピュータ４から、Ｊａｖａ実行環境の種類が異なる複数のクライアントコンピュータ１～３に対して、各クライアントコンピュータ１～３におけるリモートメソッド呼び出しに使えるスタブクラスを動的に生成してダウンロードし、同じサーバオブジェクトのメソッドに対するリモートメソッド呼び出しを可能にしている。このため、構成が様々に異なるクライアントコンピュータ１～３に対して同じ入口のサービスを一つだけサーバコンピュータに用意しておくだけで済ますことができ、また、スタブクラスをサーバコンピュータ４に保存しておく必要がなくなる。

【００７２】

【発明の効果】

以上説明したように本発明によれば、１つのサーバコンピュータから、実行環境の種類が異なる複数のクライアントコンピュータに対して適切なスタブをダウンロードすることが可能となる。また、スタブ生成部によってスタブを動的に生成する構成にあっては、予め全ての構成に対応するスタブをサーバ側に用意しておく必要がなくなる。

【図面の簡単な説明】

【図１】

本発明の一実施の形態にかかるコンピュータシステムのブロック図である。

【図２】

サーバコンピュータに保持するスタブクラスを生成する構成例を示すブロック図である。

【図３】

サーバコンピュータに保持するスタブクラスを生成する別の構成例を示すブロック図である。

【図４】

本発明の一実施の形態にかかるコンピュータシステムにおけるＲＭＩの処理手順を示すフローチャートである。

【図５】

本発明の別の実施の形態にかかるコンピュータシステムのブロック図である。

【図６】

スタブ生成部の構成例を示すブロック図である。

【図７】

本発明の別の実施の形態にかかるコンピュータシステムにおけるＲＭＩの処理手順を示すフローチャートである。

【図８】

本発明の更に別の実施の形態にかかるコンピュータシステムのブロック図である。

【図９】

本発明の更に別の実施の形態にかかるコンピュータシステムにおけるＲＭＩの処理手順を示すフローチャートである。

【図１０】

ＲＭＩの仕組みの概要を示す図である。

【図１１】

従来のコンピュータシステムのブロック図である。

【図１２】

従来のコンピュータシステムにおけるＲＭＩの処理手順を示すフローチャートである。

【符号の説明】

１、２、３…クライアントコンピュータ

４…サーバコンピュータ

５…ネームサーバコンピュータ

６…ネットワーク

１０、１６、１７、２０…Ｊａｖａ実行環境

１１、２１…クラス

１２、２３…制御部

１３…スタブ検索部

１４、２４…クラスローダ

１５、２５…インスタンス

２２…スケルトンクラス

２６－１～２６－３…スタブクラス

２７…スタブ検索インターフェース

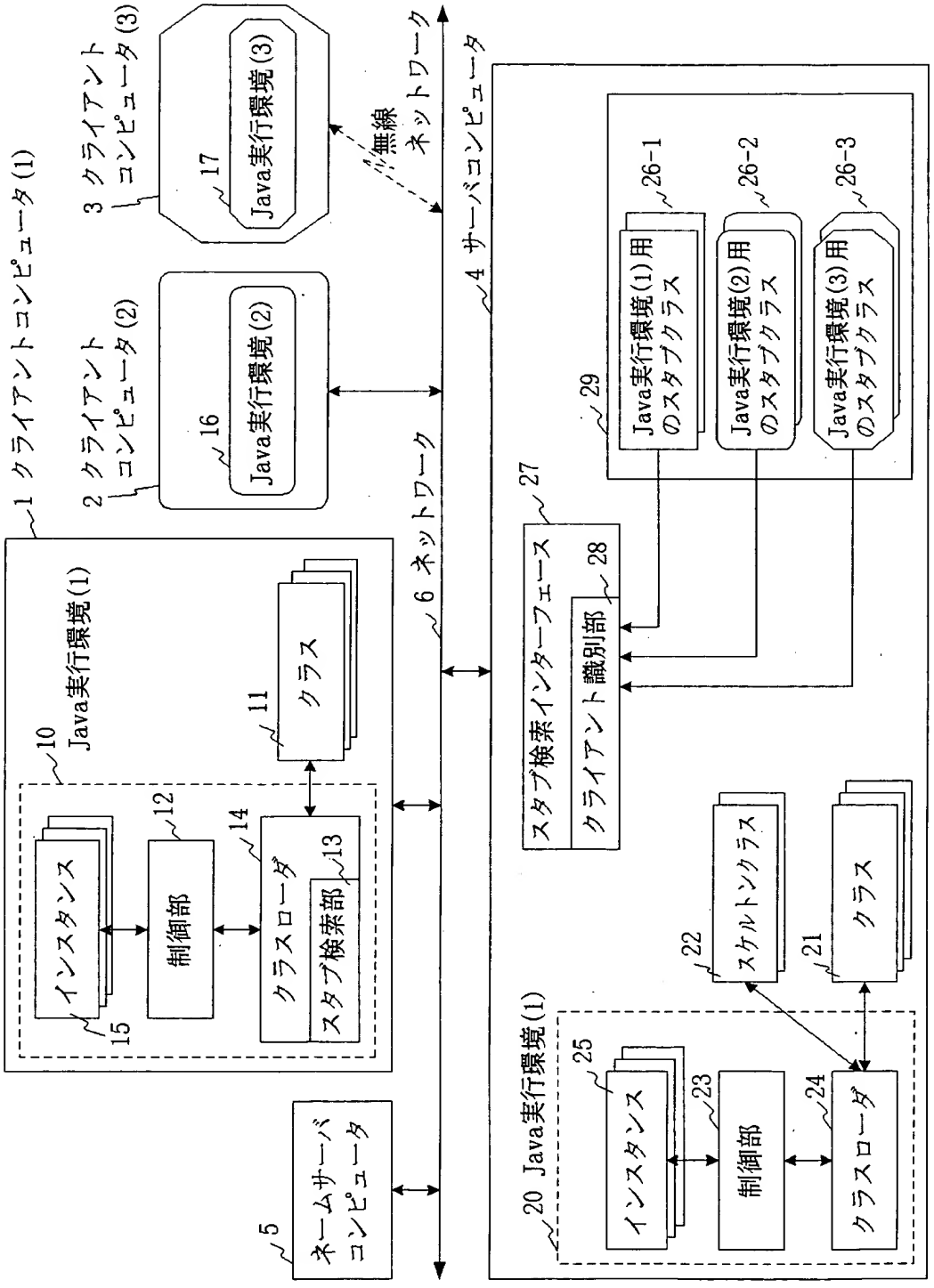
２８…クライアント識別部

２９…記憶部

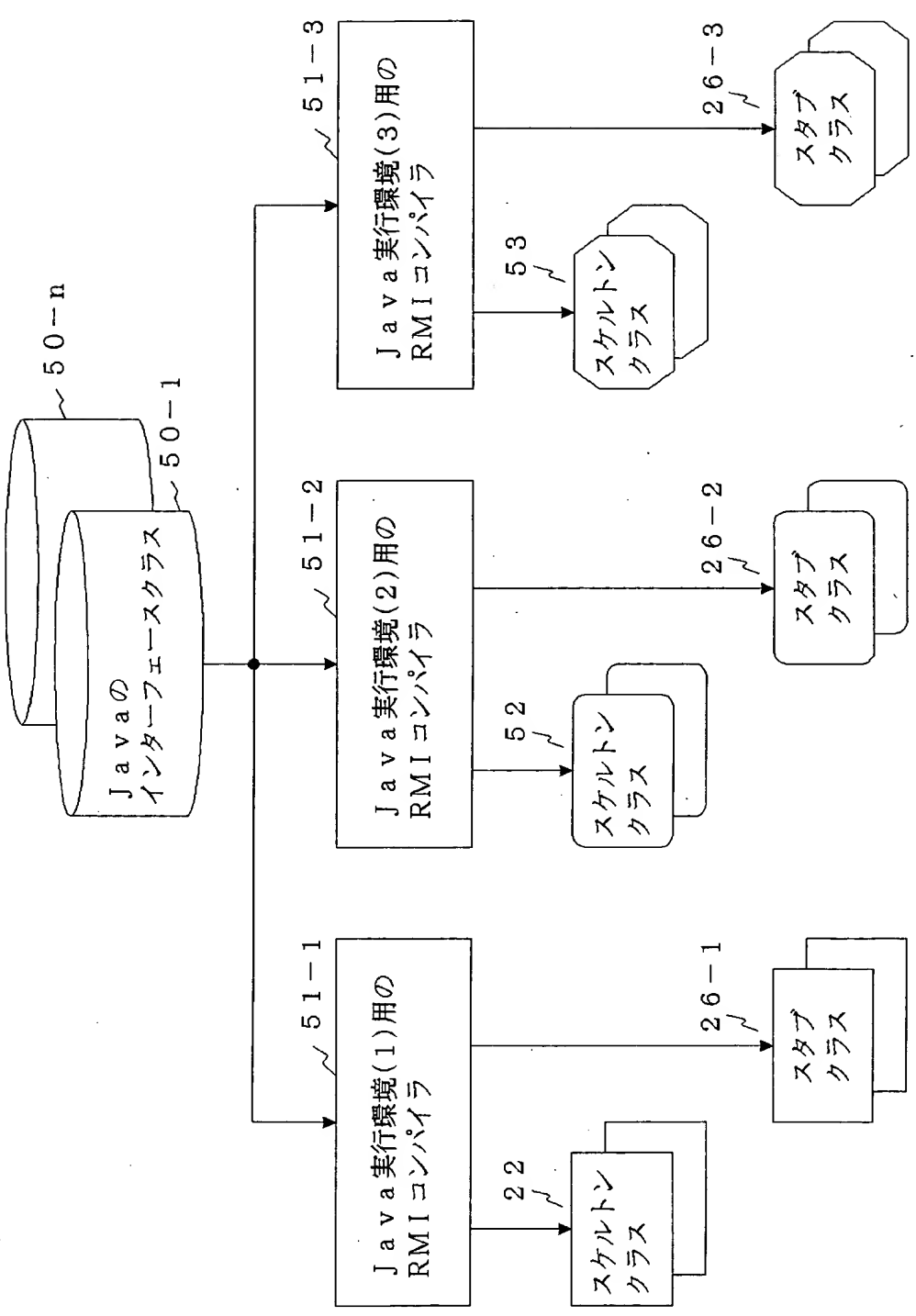
３０…スタブ生成部

【書類名】　　図面

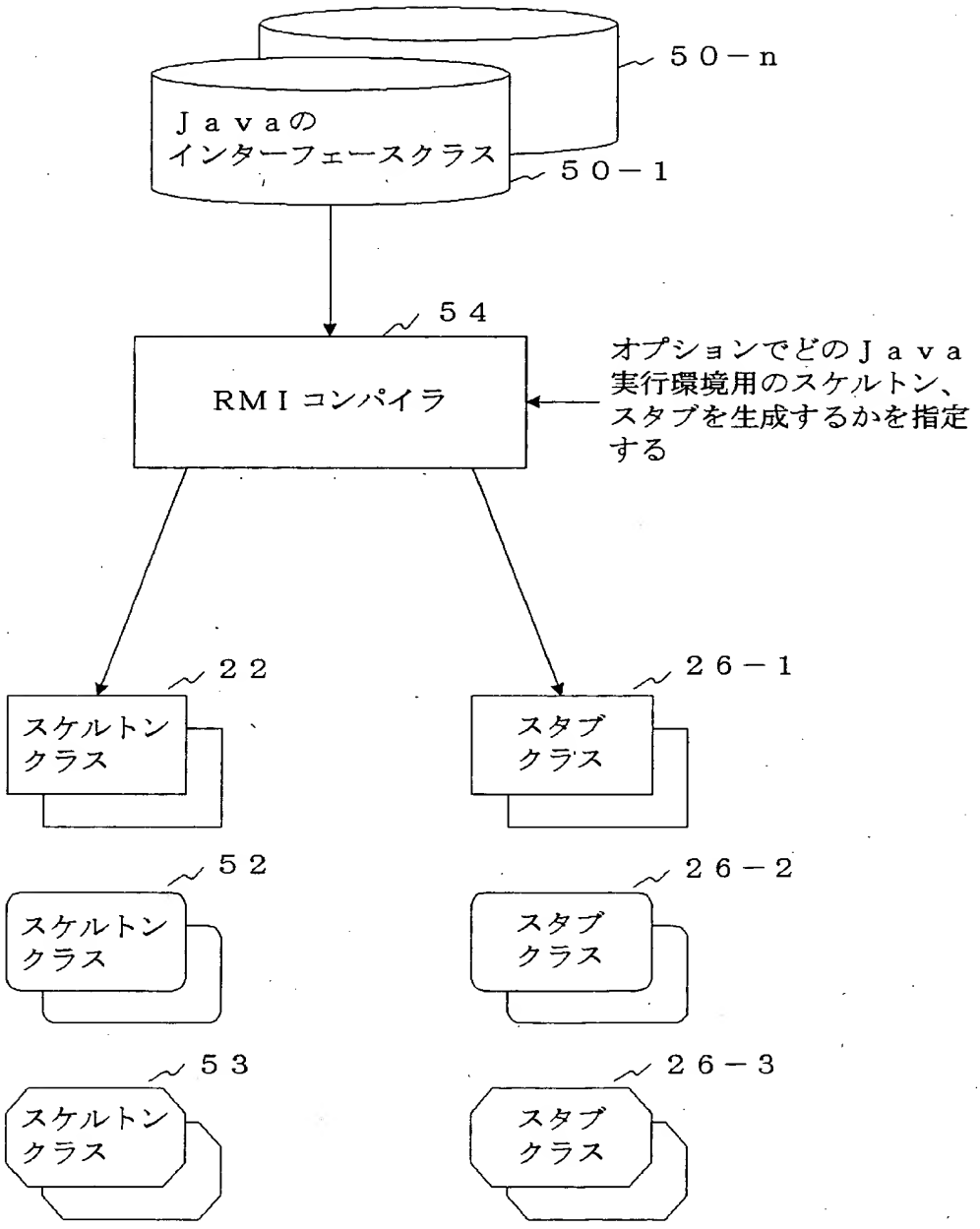
【図１】


1 クライアントコンピュータ(1)
2 クライアント コンピュータ(2)
3 クライアント コンピュータ(3)
4 サーバコンピュータ
5 ネームサーバ コンピュータ
6 ネットワーク
10 Java実行環境(1)
11 クラス
12 制御部
13 スタブ検索部
14 クラスローダ
15 インスタンス
16 Java実行環境(2)
17 Java実行環境(3)
無線 ネットワーク
20 Java実行環境(1)
21 クラス
22 スケルトンクラス
23 制御部
24 クラスローダ
25 インスタンス
26-1 Java実行環境(1)用のスタブクラス
26-2 Java実行環境(2)用のスタブクラス
26-3 Java実行環境(3)用のスタブクラス
27 スタブ検索インターフェース
28 クライアント識別部
29

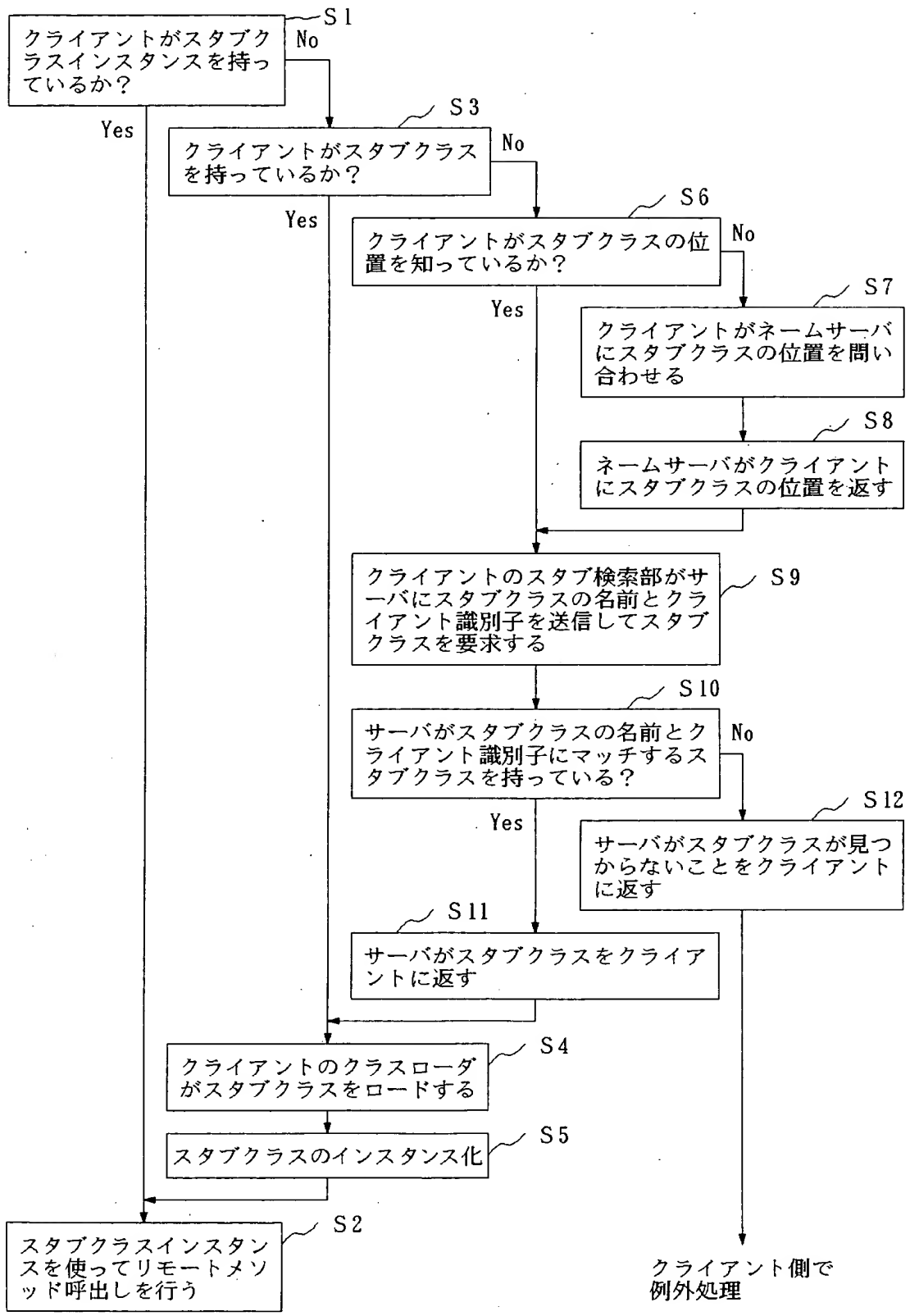
【図２】


５０－ｎ
５０－１
Ｊａｖａの
インターフェースクラス
５１－１
Ｊａｖａ実行環境(１)用の
ＲＭＩコンパイラ
５１－２
Ｊａｖａ実行環境(２)用の
ＲＭＩコンパイラ
５１－３
Ｊａｖａ実行環境(３)用の
ＲＭＩコンパイラ
２２
スケルトン
クラス
５２
スケルトン
クラス
５３
スケルトン
クラス
２６－１
スタブ
クラス
２６－２
スタブ
クラス
２６－３
スタブ
クラス

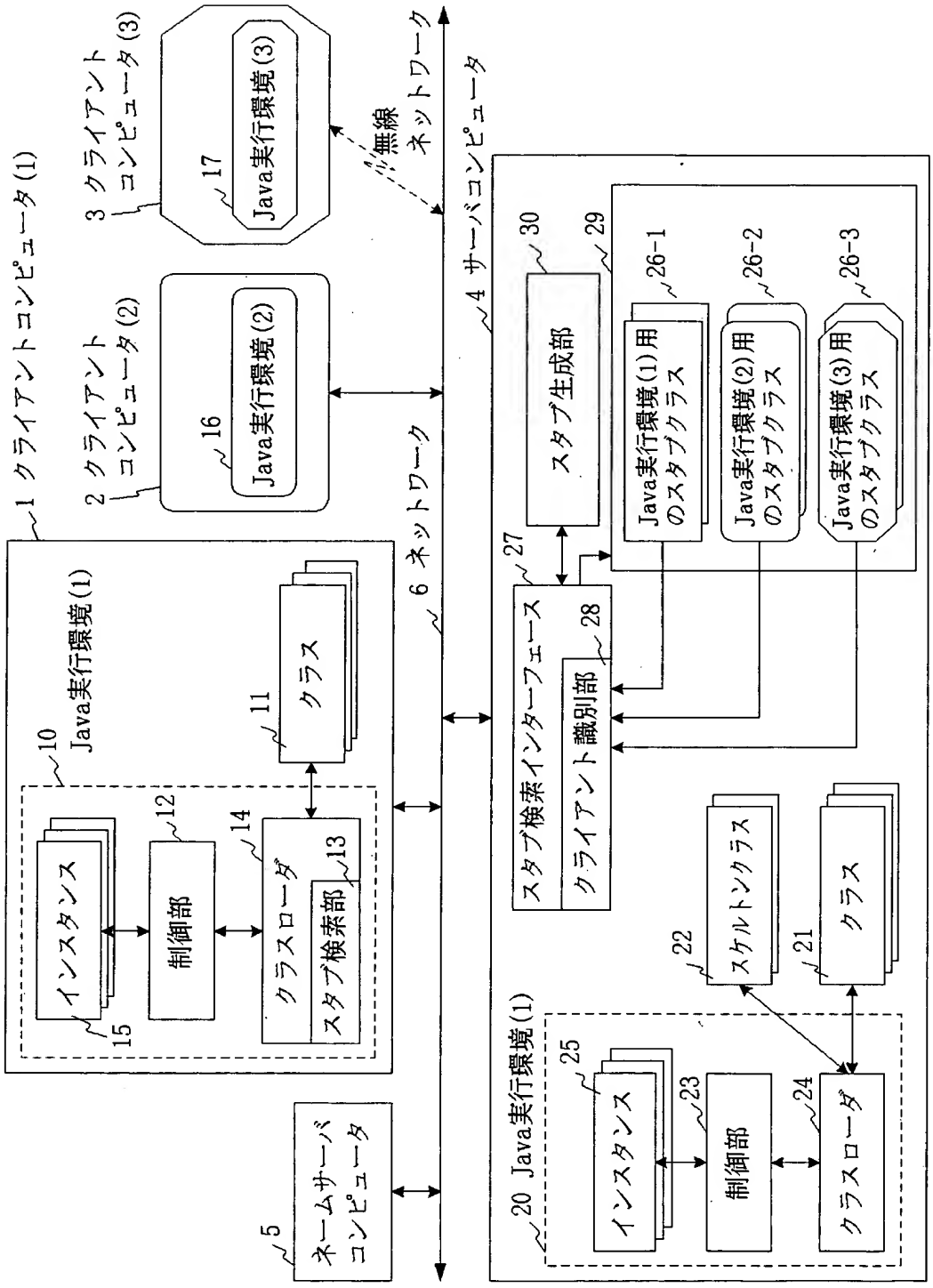
【図３】


５０－ｎ
Ｊａｖａの
インターフェースクラス
５０－１
５４
ＲＭＩコンパイラ
オプションでどのＪａｖａ
実行環境用のスケルトン、
スタブを生成するかを指定
する
２２
スケルトン
クラス
２６－１
スタブ
クラス
５２
スケルトン
クラス
２６－２
スタブ
クラス
５３
スケルトン
クラス
２６－３
スタブ
クラス

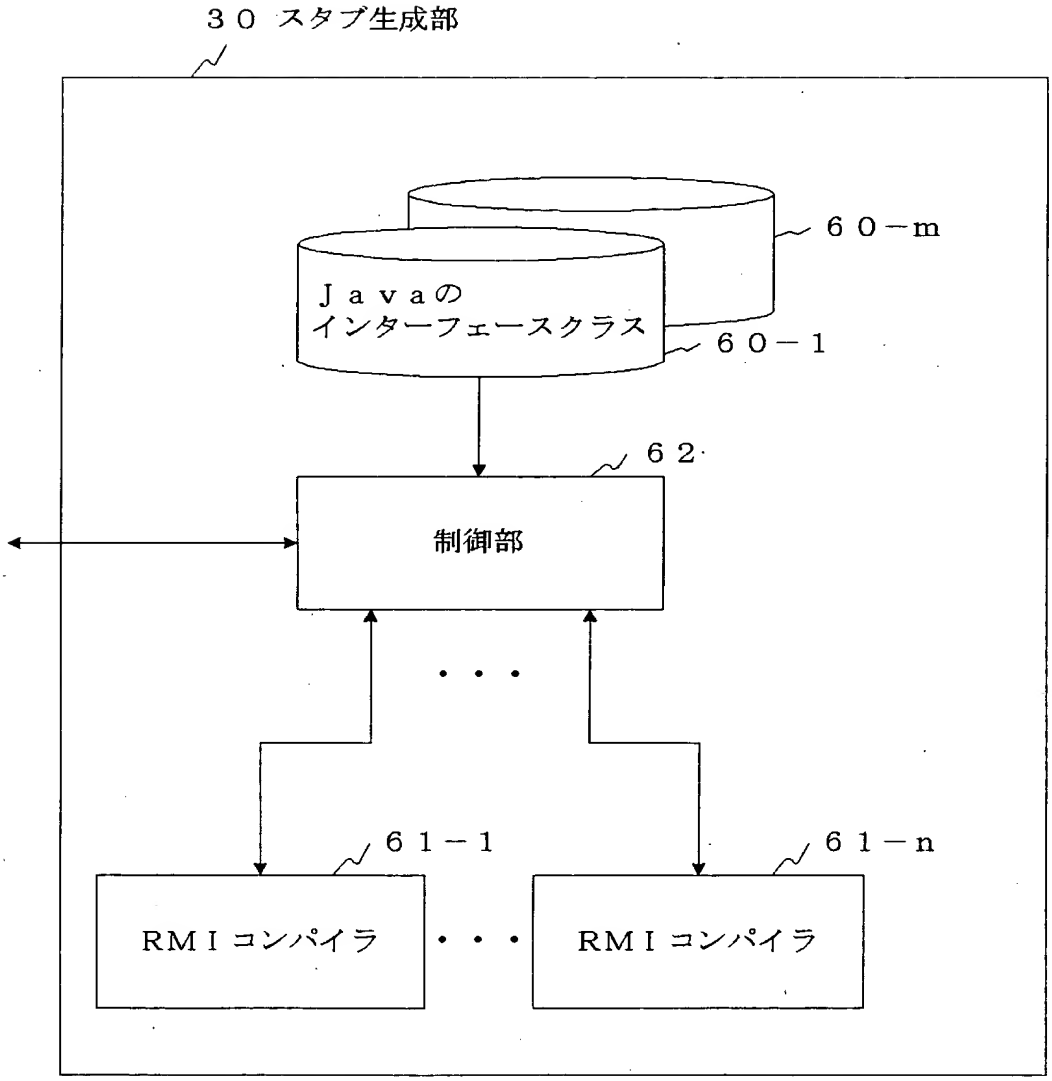
【図４】


S1
クライアントがスタブクラスインスタンスを持っているか？
No
Yes
S3
クライアントがスタブクラスを持っているか？
No
Yes
S6
クライアントがスタブクラスの位置を知っているか？
No
Yes
S7
クライアントがネームサーバにスタブクラスの位置を問い合わせる
S8
ネームサーバがクライアントにスタブクラスの位置を返す
クライアントのスタブ検索部がサーバにスタブクラスの名前とクライアント識別子を送信してスタブクラスを要求する
S9
S10
サーバがスタブクラスの名前とクライアント識別子にマッチするスタブクラスを持っている？
No
Yes
S12
サーバがスタブクラスが見つからないことをクライアントに返す
S11
サーバがスタブクラスをクライアントに返す
S4
クライアントのクラスローダがスタブクラスをロードする
S5
スタブクラスのインスタンス化
S2
スタブクラスインスタンスを使ってリモートメソッド呼出しを行う
クライアント側で例外処理

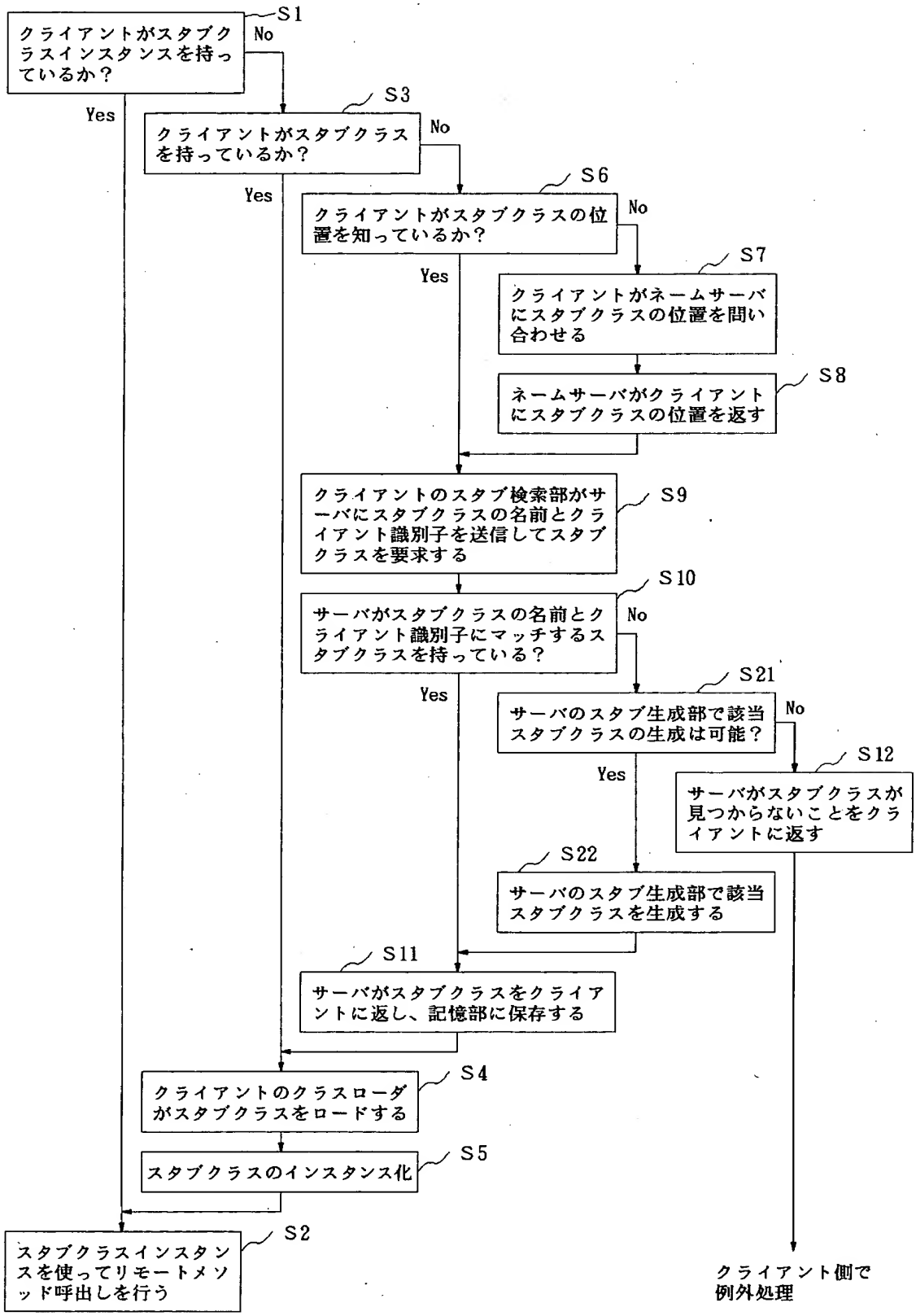
【図５】


1 クライアントコンピュータ(1)
2 クライアント コンピュータ(2)
3 クライアント コンピュータ(3)
4 サーバコンピュータ
5 ネームサーバ コンピュータ
6 ネットワーク
無線 ネットワーク
10 Java実行環境(1)
11 クラス
12 制御部
13 スタブ検索部
14 クラスローダ
15 インスタンス
16 Java実行環境(2)
17 Java実行環境(3)
20 Java実行環境(1)
21 クラス
22 スケルトンクラス
23 制御部
24 クラスローダ
25 インスタンス
26-1 Java実行環境(1)用 のスタブクラス
26-2 Java実行環境(2)用 のスタブクラス
26-3 Java実行環境(3)用 のスタブクラス
27 スタブ検索インターフェース
28 クライアント識別部
29 スタブ生成部
30

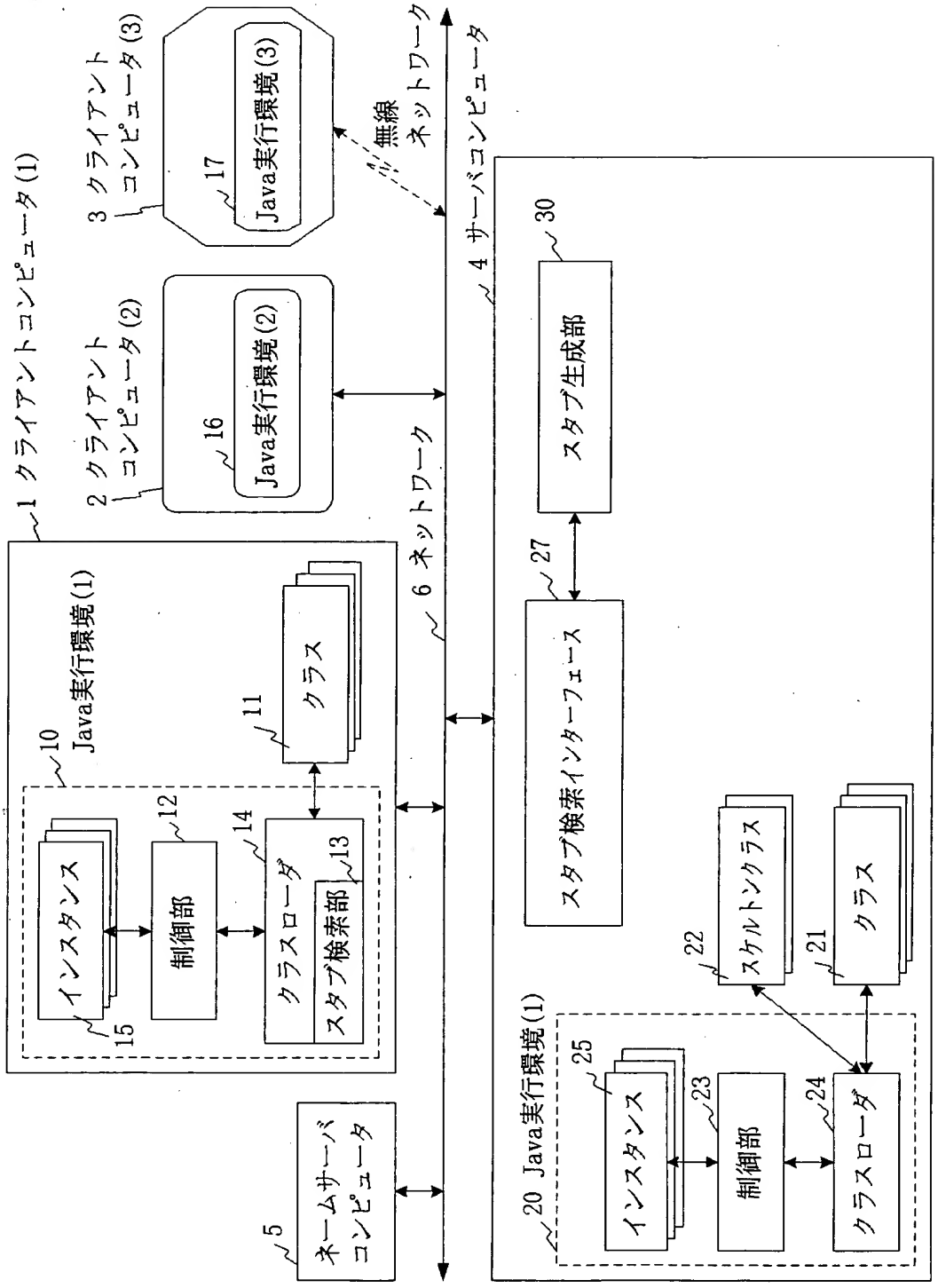
【図６】


３０ スタブ生成部
６０－m
Ｊａｖａの
インターフェースクラス
６０－１
６２
制御部
・・・
６１－１
６１－n
RMIコンパイラ
・・・
RMIコンパイラ

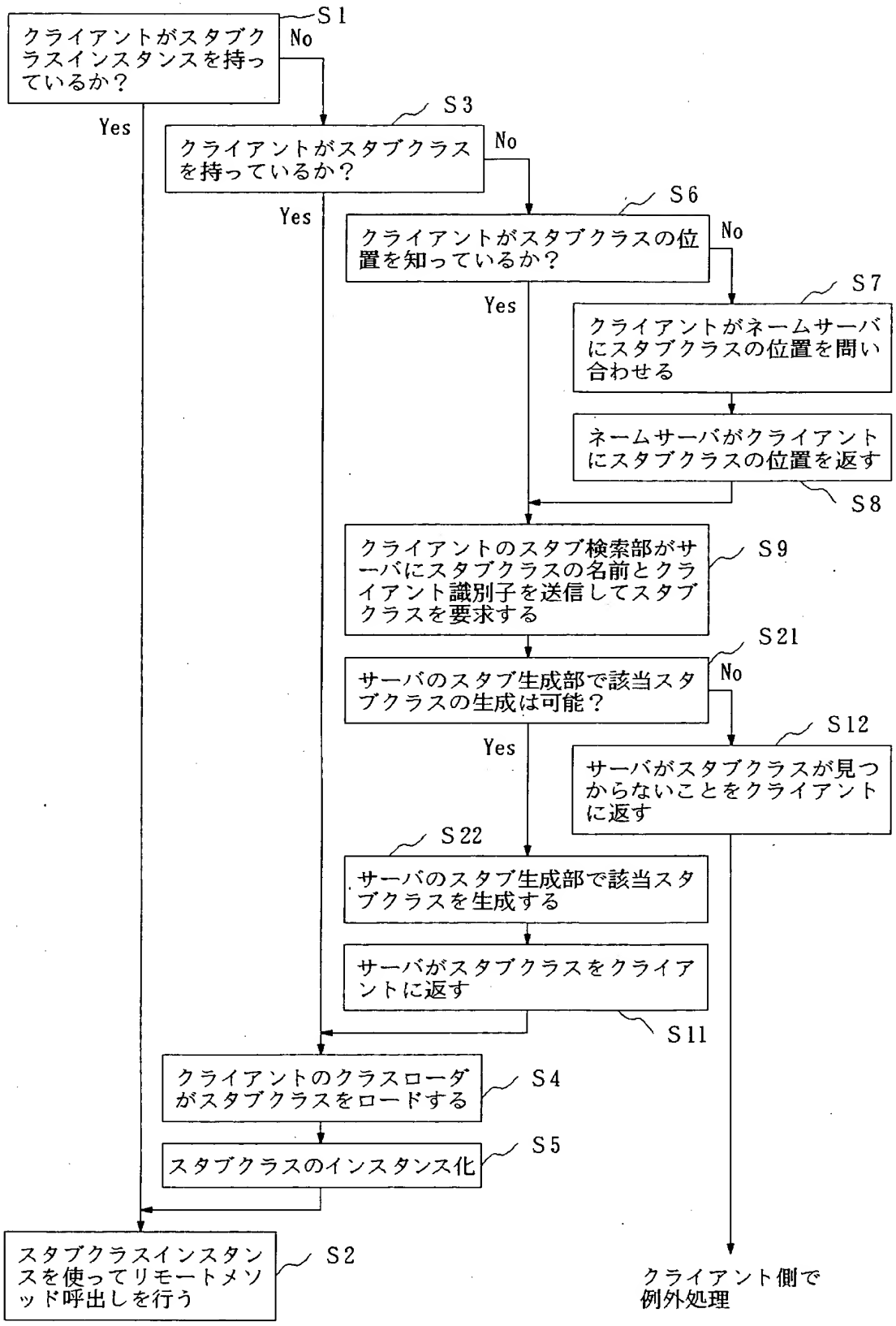
【図７】


S1
クライアントがスタブクラスインスタンスを持っているか？
No
Yes
S3
クライアントがスタブクラスを持っているか？
No
Yes
S6
クライアントがスタブクラスの位置を知っているか？
No
Yes
S7
クライアントがネームサーバにスタブクラスの位置を問い合わせる
S8
ネームサーバがクライアントにスタブクラスの位置を返す
S9
クライアントのスタブ検索部がサーバにスタブクラスの名前とクライアント識別子を送信してスタブクラスを要求する
S10
サーバがスタブクラスの名前とクライアント識別子にマッチするスタブクラスを持っている？
No
Yes
S21
サーバのスタブ生成部で該当スタブクラスの生成は可能？
No
Yes
S12
サーバがスタブクラスが見つからないことをクライアントに返す
S22
サーバのスタブ生成部で該当スタブクラスを生成する
S11
サーバがスタブクラスをクライアントに返し、記憶部に保存する
S4
クライアントのクラスローダがスタブクラスをロードする
S5
スタブクラスのインスタンス化
S2
スタブクラスインスタンスを使ってリモートメソッド呼出しを行う
クライアント側で例外処理

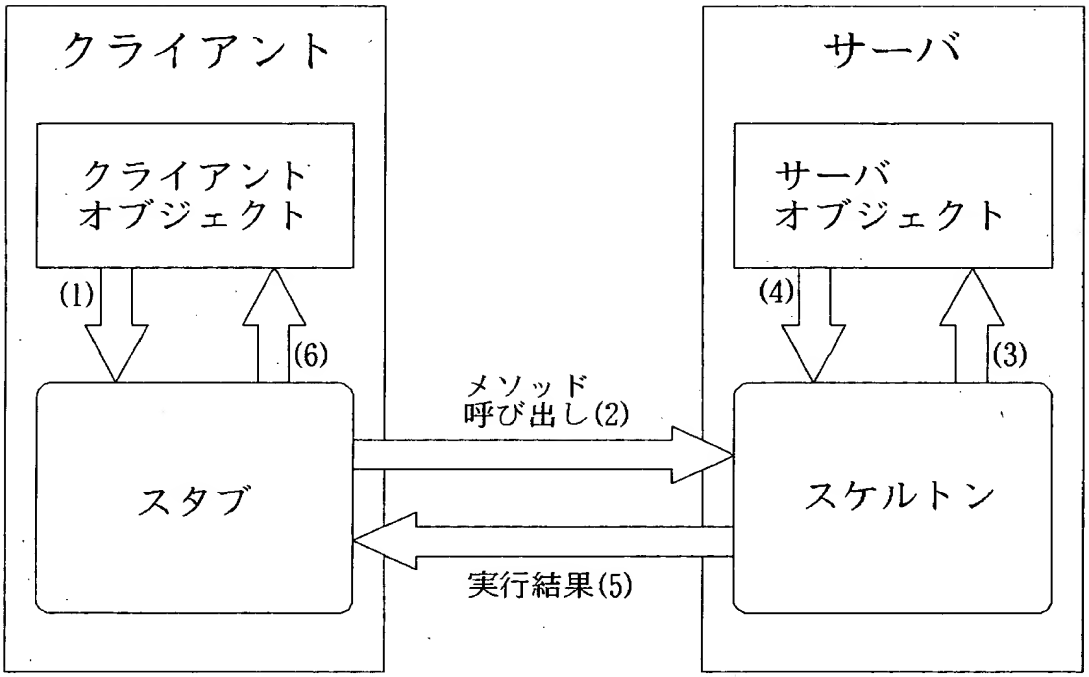
【図８】


1 クライアントコンピュータ(1)
2 クライアント コンピュータ(2)
3 クライアント コンピュータ(3)
4 サーバコンピュータ
5 ネームサーバ コンピュータ
6 ネットワーク
無線 ネットワーク
10 Java実行環境(1)
11 クラス
12 制御部
13 スタブ検索部
14 クラスローダ
15 インスタンス
16 Java実行環境(2)
17 Java実行環境(3)
20 Java実行環境(1)
21 クラス
22 スケルトンクラス
23 制御部
24 クラスローダ
25 インスタンス
27 スタブ検索インターフェース
30 スタブ生成部

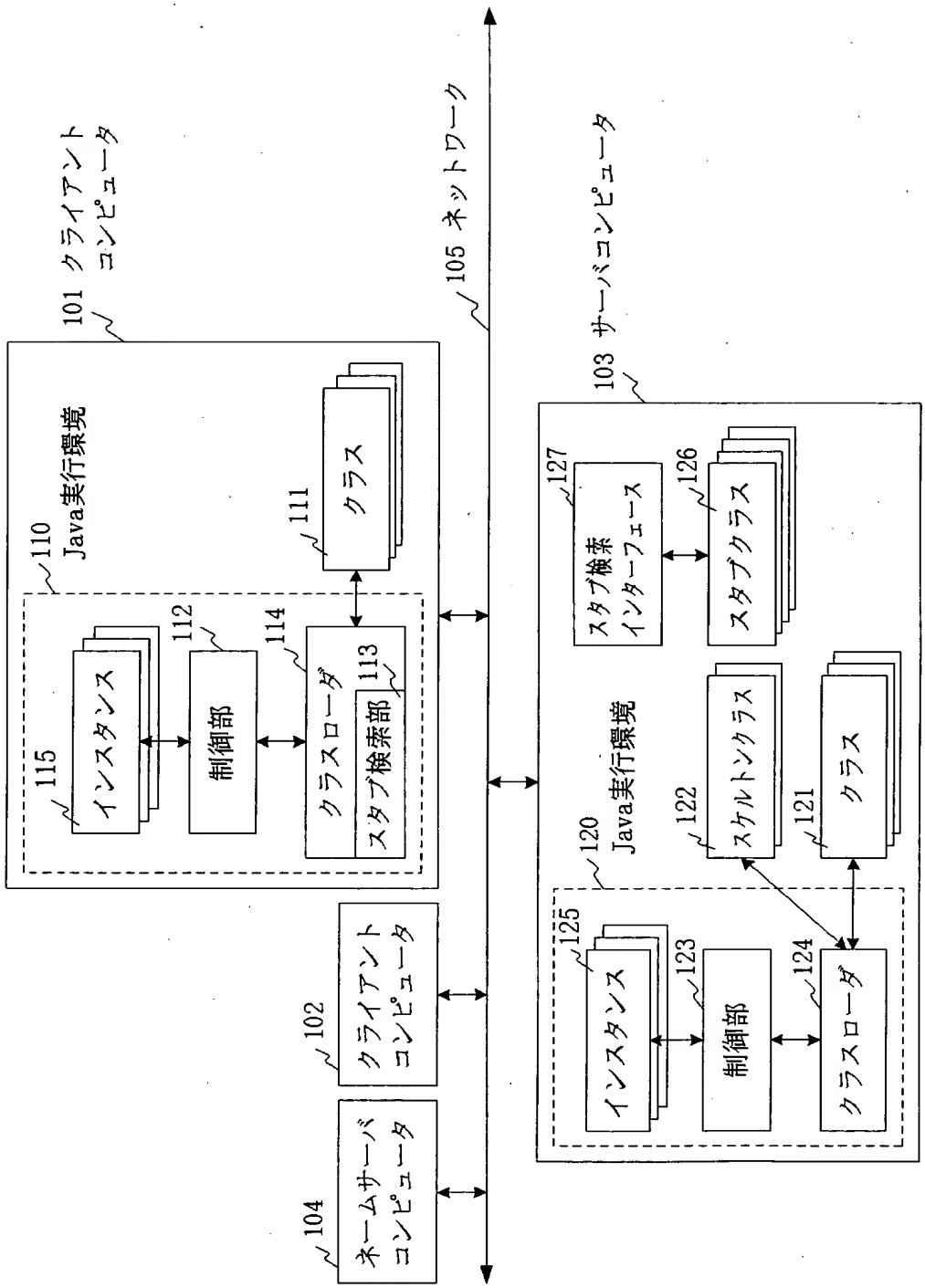
【図９】


S1
クライアントがスタブクラスインスタンスを持っているか？
No
Yes
S3
クライアントがスタブクラスを持っているか？
No
Yes
S6
クライアントがスタブクラスの位置を知っているか？
No
Yes
S7
クライアントがネームサーバにスタブクラスの位置を問い合わせる
ネームサーバがクライアントにスタブクラスの位置を返す
S8
S9
クライアントのスタブ検索部がサーバにスタブクラスの名前とクライアント識別子を送信してスタブクラスを要求する
S21
サーバのスタブ生成部で該当スタブクラスの生成は可能？
No
Yes
S12
サーバがスタブクラスが見つからないことをクライアントに返す
S22
サーバのスタブ生成部で該当スタブクラスを生成する
サーバがスタブクラスをクライアントに返す
S11
S4
クライアントのクラスローダがスタブクラスをロードする
S5
スタブクラスのインスタンス化
S2
スタブクラスインスタンスを使ってリモートメソッド呼出しを行う
クライアント側で例外処理

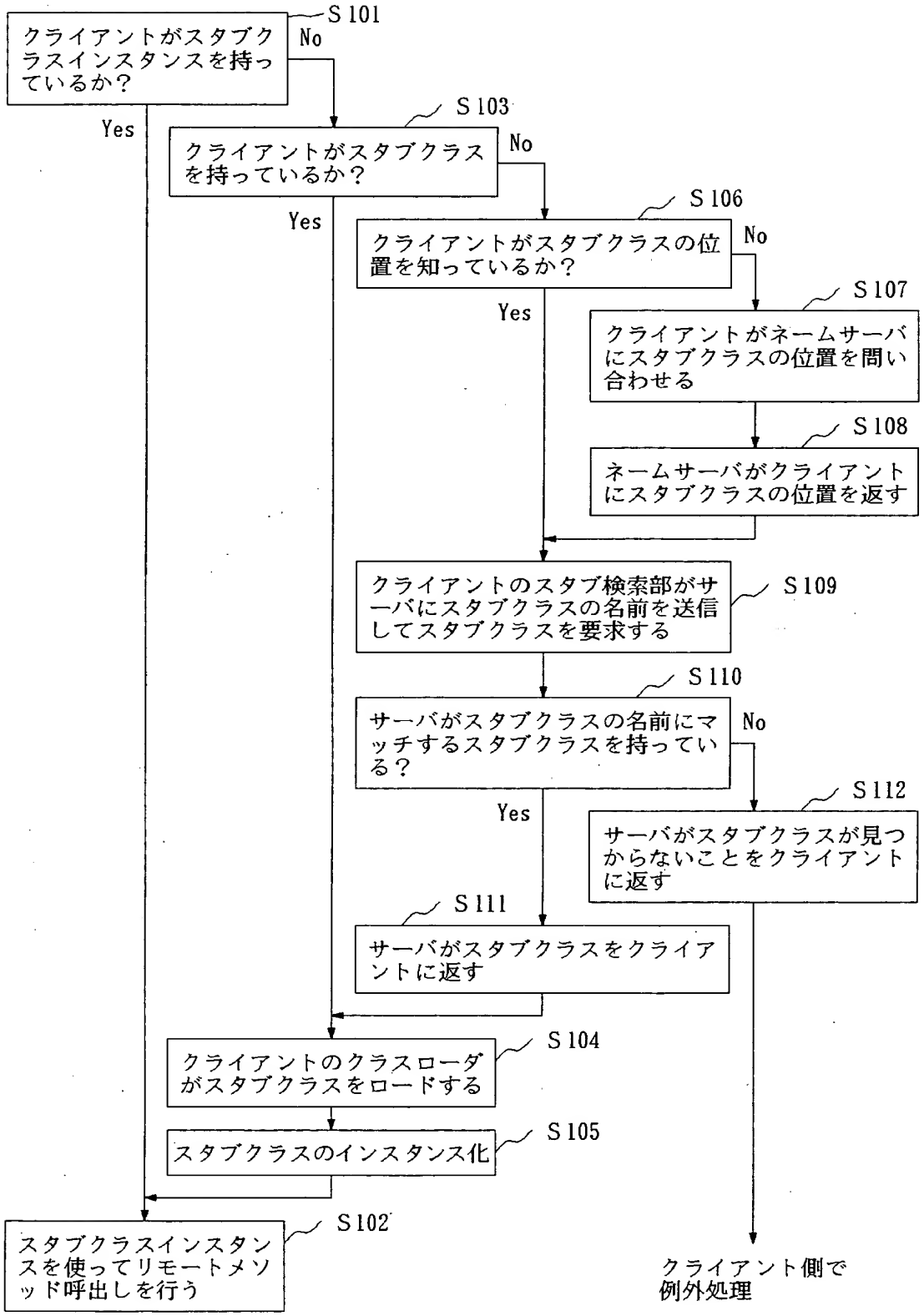
【図１０】


クライアント
サーバ
クライアント
オブジェクト
サーバ
オブジェクト
(1)
(6)
(4)
(3)
メソッド
呼び出し(2)
スタブ
スケルトン
実行結果(5)

【図１１】


101 クライアント コンピュータ
105 ネットワーク
103 サーバコンピュータ
110 Java実行環境
111 クラス
112 制御部
113 スタブ検索部
114 クラスローダ
115 インスタンス
127 スタブ検索 インターフェース
126 スタブクラス
120 Java実行環境
122 スケルトンクラス
121 クラス
125 インスタンス
123 制御部
124 クラスローダ
102 クライアント コンピュータ
104 ネームサーバ コンピュータ

【図１２】


S101
クライアントがスタブクラスインスタンスを持っているか？
No
Yes
S103
クライアントがスタブクラスを持っているか？
No
Yes
S106
クライアントがスタブクラスの位置を知っているか？
No
Yes
S107
クライアントがネームサーバにスタブクラスの位置を問い合わせる
S108
ネームサーバがクライアントにスタブクラスの位置を返す
クライアントのスタブ検索部がサーバにスタブクラスの名前を送信してスタブクラスを要求する
S109
S110
サーバがスタブクラスの名前にマッチするスタブクラスを持っている？
No
Yes
S112
サーバがスタブクラスが見つからないことをクライアントに返す
S111
サーバがスタブクラスをクライアントに返す
S104
クライアントのクラスローダがスタブクラスをロードする
S105
スタブクラスのインスタンス化
S102
スタブクラスインスタンスを使ってリモートメソッド呼出しを行う
クライアント側で例外処理

【書類名】　　要約書

【要約】

【課題】　１つのサーバコンピュータから、Ｊａｖａ実行環境の種類が異なる複数のクライアントコンピュータに対して、各クライアントコンピュータにおけるリモートメソッド呼び出しに使えるスタブクラスをダウンロードできるようにする。

【解決手段】　サーバコンピュータ４には、スタブを実行環境の異なるクライアント１～３の種類毎に用意したスタブクラス２６－１～２６－３がある。各クライアントコンピュータ１～３はスタブが必要になったとき、スタブ名とクライアント識別子とを指定したスタブ要求をネットワーク６経由でサーバコンピュータ４に送出する。サーバコンピュータ４のスタブ検索インターフェース２７は、指定されたスタブ名とクライアント識別子に基づいて適切なスタブをスタブクラス２６－１～２６－３から検索して要求元のクライアントに返す。

【選択図】　　図１

出　願　人　履　歴　情　報

識別番号　　〔０００００４２３７〕

１．変更年月日　　１９９０年　８月２９日
［変更理由］　新規登録
住　所　東京都港区芝五丁目７番１号
氏　名　日本電気株式会社